Intelligent Information Filtering via Hybrid Techniques: Hill Climbing, Case-Based Reasoning, Index Patterns, and Genetic Algorithms

BY

Kenrick Jefferson Mock

B.S. (University of California, Davis) 1990

M.S. (University of California, Davis) 1994

DISSERTATION

Submitted in partial satisfaction of the requirements for the degree of

DOCTOR OF PHILOSOPHY

in

COMPUTER SCIENCE

In the

OFFICE OF GRADUATE STUDIES

of the

UNIVERSITY OF CALIFORNIA

DAVIS

Approved:

_____

_____

_____

Committee in Charge

1996

Intelligent Information Filtering via Hybrid Techniques: Hill Climbing, Case-Based Reasoning, Index Patterns, and Genetic Algorithms

Abstract

As the size of the Internet increases, the amount of data available to users has dramatically risen, resulting in an information overload for users. This work shows that information overload is a problem, and that data is organized poorly by existing browsers. To address these problems, an intelligent information news filtering system named INFOS (Intelligent News Filtering Organizational System) was created to reduce the user's search burden by automatically eliminating Usenet news articles predicted to be irrelevant. These predictions are learned automatically by adapting an internal user model that is based upon features taken from articles and collaborative features derived from other users. The features are manipulated through keyword-based techniques, knowledge-based techniques, and genetic algorithms to build a user model to perform the actual filtering. The integration of knowledge-based techniques for in-depth analysis, statistical and keyword approaches for scalability, and genetic algorithms for exploration allows INFOS to achieve better filtering performance than by using either technique alone. Experimental results collected from the prototype of INFOS validate the gain in performance within the domain of news articles posted to electronic bulletin boards.

Acknowledgments

Throughout the course of this project, I have interacted with many people that have influenced the development of my professional work as well as my personal growth. I would like to thank the following people:

My advisor, Sergio Alvarado, for his guidance and support throughout my years at Davis. His patience and expertise helped me find direction in many endeavors. I found the research projects to be challenging, exciting, and worthy to undertake. Without his mentoring, ability to excite me about AI, and encouragement to study in the field, I would probably have never returned to graduate school.

My other committee members, Dick Walters and Rao Vemuri. Dick's useful comments has been particularly helpful in improving my writing style. Rao's mentoring, willingness to let me explore areas of research, and introduction to exciting areas of research have always been appreciated.

The many volunteer subjects, both UC Davis students and WWW surfers, who spent long hours testing the system.

The members of the AI lab (and other research labs), from the old gang of Ron Braun, Doug Mayfield, Evan Fletcher, Katia Chiriatti, Jeremy Frank and Robert Davis to the newer gang of Keith Wear, Thuyen Nguyen, Jennifer Dunham, and Nancy Reed. All of them provided companionship and kept work fun in addition to providing valuable scholarly input.

All of my IRC friends, especially Joe, Jessica, Iba, Bobo, Eddie, Rodney, Hal, Susan, Subash, Laura, Ilene, Kearston, Debbie, Leora, Mary, Mike, and Donna. They certainly helped me procrastinate, but made it a lot of fun along the way.

UC Micro and Apple Computer, including Rao Machiraju, Mike Graves, and Rick Borovoy for providing financial support for the original work and many useful discussions.

Michelle Hoyle, for being there when I needed technical or emotional support.

Khristy Parker, for providing unrelenting support down the stretch along with much-needed inspiration, code, magic, and truffles.

Finally, I would like to thank my family. My sister Theresa and brother Cary provided much needed motivation, while my loving parents Jeff and Lonnie made all of this possible through their support, education, and patience.

Table of Contents

List of Figures

List of Tables

1. Introduction

As networked systems grow in size, the amount of data available to users has increased dramatically. The result is an information overload for the user. This project has investigated the uses of an intelligent information filtering system named INFOS (Intelligent News Filtering Organizational System) to reduce the user's search burden by automatically eliminating data predicted to be irrelevant. Unlike the majority of news readers that require users to explicitly create a user profile to perform filtering, INFOS is capable of learning this profile automatically. These predictions are learned by adapting an internal user model that is based upon user interactions and collaborative actions of other users. The primary domain for the project is the filtering of Usenet news articles.

1.1 The Information Overload Problem

With the advent of networked systems, computer users are inundated with information that they cannot efficiently utilize. Tools are urgently needed to assist the user with information filtering devices in order to reduce the user's search burden. This project has examined the Usenet News system as a testbed for the filtering algorithms. In the Usenet system, users throughout the world intermittently post articles to a common bulletin board. The number of articles posted may be very large; e.g., newsgroups may receive hundreds of articles daily. These messages are queued temporarily, and removed in a FIFO manner. Note that there is a constant throughput of messages, as new messages replace old messages. Consequently, the lifetime of a message may be anywhere from a week to several months, depending on the amount of traffic. Nevertheless, there is a huge amount of data in this message stream. In the UC Davis news spool, over 1.4 gigabytes of disk space is required to store news messages at any time. These messages

are spread out across approximately 5000 newsgroups, resulting in approximately 38,000 new messages delivered each day. The large volume of data makes it extremely difficult for a user to extract useful information.

The goal of this project is to predict whether new articles are likely to be of interest, or not of interest, based upon the prior behavior of the user. Systems that perform this type of intelligent behavior have recently been touted as intelligent "agents" (Riecken, 1994) by the media. The work proposed here follows the same vein; the system is intended to aid the user in his or her work rather than take over completely. The system must work beside the user like an aide or agent, watching and learning what the user does and what the user is interested in so that intelligent filtering may be performed. The filtering task is an extremely fuzzy and difficult problem to solve since users are notorious for their inconsistencies in behavior and interests. From a machine learning perspective, the problem is similar to trying to approximate a curve based upon discrete data points - except in this case, the function the machine is trying to approximate may change at any time.

One of the difficult constraints imposed by this type of problem is the necessity for dealing with change. Many learning algorithms require repeated training epochs over a fixed data set. In the Usenet News problem, the data set is constantly changing as incoming messages are posted. To ensure consistency the method would need to store all messages ever posted. This is clearly undesirable due to the time requirements for training and the space required to store all messages. Most approaches to the information filtering problem bypass this problem by forcing the user to define explicitly what should be filtered, e.g., via a *keyword* based database language (Goldberg, 1992). Keywords or *tokens* are simply words or lexical items devoid of any of the semantic information regarding those tokens.

Although keyword based systems have been popular due to their simplicity, the performance of keyword systems ultimately suffers due to the "keyword barrier." As

described by Mauldin (1991), the keyword barrier arises since keyword systems do not actually understand the semantic content of the input articles. On the other hand, understanding systems consider semantic content by processing articles in a manner similar to humans. Understanding systems can break the keyword barrier and achieve higher performance. A middle ground can be achieved through a hybrid system that incorporates keywords and limited semantic knowledge. The expected performance of a hybrid system is shown in figure 1 along with the expected performance of keyword and understanding systems as projected by Mauldin.

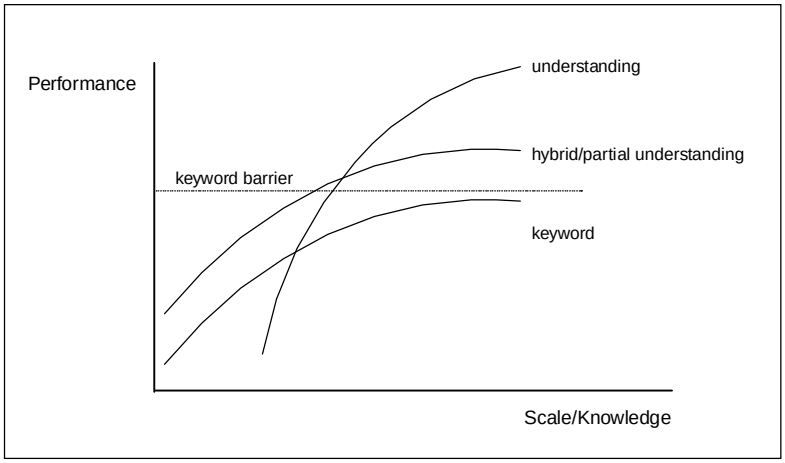


Figure 1 : Expected System Performance vs. Scale/ Input Knowledge
for Keyword-Based, Understanding-Based, and Hybrid-Based systems.

Figure 1 indicates that understanding-based systems perform best in the long run. However, the increase in performance is accompanied with a much larger cost (time, system resources, complexity) than the other systems. A large portion of the cost is comprised of the massive volume of hand-coded knowledge that must be input by knowledge engineers. In addition to the time required for human input, as more knowledge is added to the system the number of relationships among pieces of knowledge increases. To handle this complexity, cognitive processes must be modeled to intelligently relate the knowledge. As a result of the large cost, it is extremely difficult to create a

successful understanding-based system that operates on a large scale. On the other hand, keyword systems do not require such intricate processes, knowledge, nor detailed human intervention. Consequently, the cost of creating keyword systems is low. Finally, a hybrid system that incorporates aspects of understanding systems with keyword systems requires a smaller cost than pure understanding systems, but a larger cost than keyword systems. The cost of all systems as knowledge increases is shown in figure 2.

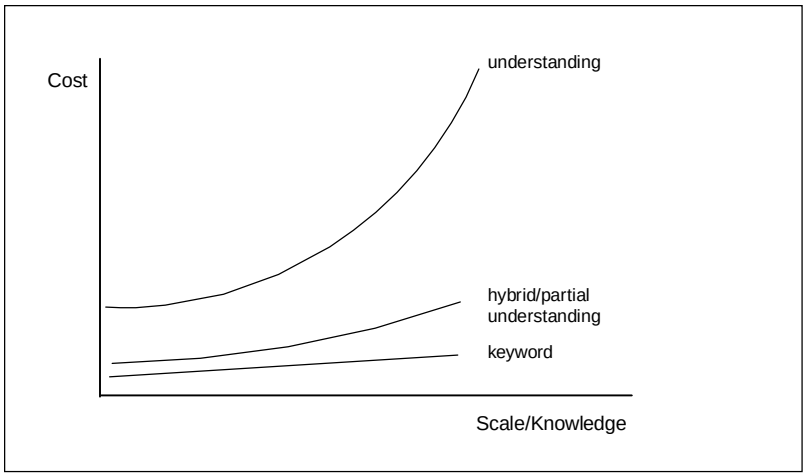


Figure 2 : Expected System Cost vs. Scale / Input Knowledge
for Keyword-Based, Understanding-Based, and Hybrid-Based systems.

This thesis focuses primarily on hybrid systems for Usenet news filtering since these types of systems remain largely unexplored. Additionally, hybrid systems are capable of providing better performance than keyword systems while also restricting cost within manageable levels. The scalability of hybrid systems combined with their improved performance over keyword systems makes them a good candidate for a powerful yet practical package that internet users can immediately utilize and modify to fit their needs.

1.2 Usenet News Background

The Usenet news system is a distributed medium where individual servers spool and forward messages throughout the network. News is broadcast with a header field attached to each message denoting the destination newsgroup along with the sender, topic, and other accounting information. Newsgroups are hierarchical. As an example, at the upper level exists the "comp." newsgroups dealing with computers, "rec." newsgroups dealing with recreation, or "alt." newsgroups dealing with alternative topics. Indexed beneath these categories are more specialized topics, such as "macintosh" or "artificial intelligence", and finally a third or more subcategorization exists to specialize the newsgroup further, such as "games" or "genetic algorithm". An example of a complete newsgroup may be denoted as "comp.ai.ga" for the computer/artificial intelligence/genetic algorithm discussion group. Note that, despite the categorization, some messages may be relevant to more than one newsgroup and messages are often posted in the "wrong" place. This problem has been addressed by Stevens through the use of "virtual" newsgroups (Stevens, 1992). Some sample newsgroups are listed in figure 3.

Newsgroup	**Title**
comp.ai	Artificial intelligence discussions.
comp.ai.genetic	Genetic algorithms in computing.
comp.ai.edu	Applications of Artificial Intelligence to Education
comp.ai.nat-lang	Natural language processing by computers.
comp.theory.info-retrieval	Information Retrieval topics (Moderated).
misc.forsale.computers.monitors	Monitors and displays for sale and wanted.
misc.forsale.computers.modems	Modems for sale and wanted.
ucd.life	Davis Chatter.
alt.fan.letterman.top-ten	Top Ten Lists from Letterman (Moderated).

Figure 3: Sample Usenet Newsgroups.
(The moderated newsgroups require a moderator to accept submitted articles before posting to the public.)

Two sample articles that may be posted to these newsgroups are shown in figure 4. The message headers indicate the author, subject, and newsgroups. The top message may be of interest to AI researchers, while the bottom message is a chain letter. Messages such as the chain letters are often targets readers wish to have filtered out. However, this is dependent upon individual user preferences, as some readers may be interested in chain letters.

Subject: AI, Genetic Algorithms and Artificial Life
From: ai@hyperion.ucdavis.edu
Newsgroups: comp.ai

>what is your work?
My latest research on an artificially evolved critter unifying all aspects
of Artificial Intelligence is available at http://phobos.cs.ucdavis.edu:8001.

Subject: Make Money FAST!
From: grog@netcom.com
Newsgroups: comp.ai, alt.life, alt.food.chocolate

This is a chain letter. Send $5 to everyone on the list, then add yourself.
Joe did and made $5,000,000. D0 NOT break this chain or a terrible fate
shall befall you! Noooo! :-)

Figure 4: Sample News Articles

As Usenet news has evolved, an electronic culture or community complete with its own social interactions has arisen with rules of etiquette, colloquialisms, and other expressions not found in the general community of written media. For example, words with the numeral "0" substituted for "O" indicate a brazen "elite" attitude, while words in all uppercase denote shouting, or sideways "smiley faces :-)" denote facetious or humorous remarks. These subtle modifications to language are referred to as paralanguage. Other examples of paralanguage include the use of a ">" to denote material from previously posted articles, or repeated letters as in "wellllll" to denote hesitation are also commonly found. An example of paralanguage is shown in the bottom message of figure 4. The "Nooo" indicates a humorous intonation not present with a simple "no", and

the smily face indicates the message is to be taken in jest. To fully understand messages, all of these factors must be taken into account. Often, new or rebellious users do not conform to these rules, and the result may be vicious “Flames” or posted arguments and offenses complaining about others. Many users regard flames as a waste of bandwidth, and are an excellent object for a filtering system to remove.

1.3 Usenet News Interface Display

In a conventional news reader, when users read messages in a newsgroup they are given a list of articles in which the author and subject are displayed. These articles are sorted by “thread.” A thread is simply an article and all replies to that article put together into a group. By grouping messages in this way, the context of an article is maintained and articles are easier to understand. Typically, users will read entire threads at once before proceeding on to the next thread. A sample of how articles from the STRN news reader are displayed is shown in figure 5. Users are shown messages in this format and must select those of interest to read. In this example, three separate threads are shown. The first thread contains only one message, while the second thread contains one post from Marvin Minsky and three replies to his post. Similarly, the third thread contains one post from Richard Ottolini and one reply.

a G Demetriou	1 >Approximate string matching
b Marvin Minsky	4 >AI Heaven
Andreas Sigg	
Andreas Sigg	
Nancy Lebovitz	
c Richard Ottolini	2 >Does AI make philosophy obsolete?
jfrenst@ibm.net	

Figure 5: Sample STRN browser screen, messages sorted by threads

While this format for displaying messages is useful for browsing, it can be difficult to find a particular message when there are hundreds, or even thousands, of messages to

browse through. Additionally, while the author and subject of a message give a good deal of information about the article, this information can sometimes be misleading since no information from the body of the article is displayed while browsing. A common phenomena is for the subject of messages to "drift" from the original subject as replies are made to the original message. Often this can result in messages containing the original subject but displaying a completely different content.

1.4 Major Issues in Information Filtering

To create an intelligent information filter, three major issues must be addressed: (1) A method to model the user's goals, actions, expertise, interests, or behavior, (2) A method to extract key defining features from the input article text or understand the content of the article, and (3) A method to classify the input text based on the defining features from the user model and input text.

An intelligent news filter must be able to distinguish articles that the user is interested in from articles that the user is not interested in. Since users have different preferences for what makes material interesting, the news filtering system must be flexible enough to model the interests of every individual user. In INFOS, the user model is automatically created as the user reads articles and provides feedback as to whether or not the articles were interesting or not interesting. If necessary, the user is able to directly edit the model created by the system.

Before a user model can be used, or even created, news articles must be understood by the system to some degree. For information filtering, incoming articles must be understood well enough so that the content can be compared with the user model to determine if there is a match. Ultimately, complete semantic understanding as a human reader would process the text is most desirable, but a lesser degree of understanding is often sufficient for the purposes of filtering. In INFOS, keywords from the body and

subject of the article, the author, reviews by other users, and an abstraction hierarchy of concepts from the article are extracted as features to determine what an article is about and whether or not the user is going to be interested in reading the article.

After a user model has been constructed and key features extracted from input articles, an algorithm is necessary to classify new articles. In information filtering, the classifier's task is to determine the interest level of a particular document with respect to the current user. INFOS employs a statistical keyword-based classification scheme along with a case-based reasoning scheme to determine an interest level for a new document. By combining both methods, INFOS gains increased precision from the case-based reasoning scheme along with the scalability of the keyword scheme. Furthermore, by applying a genetic algorithm to a population of user models, INFOS is capable of exploring the virtual space of news articles to find other articles which may be interesting.

1.5 Overview of INFOS Architecture

To filter news articles, INFOS first presents the user with unread news articles. The user is allowed to browse these articles, and is then prompted to classify those articles that are read as being interesting, not interesting, or unknown. After new messages are read, two user models are created. The *global hill climbing table* is a model consisting of keywords pulled from the articles, the author of the article, collaborative reviews from other users, and statistics regarding the frequency of these features and the three classifications. This data is used for keyword classification of new articles. The *case-based abstraction hierarchy* is a model consisting of the semantic meaning of words that INFOS determines to be relevant to the news articles along with the classification of those articles. This data is used to classify new articles at the concept level. The computing resources required to manage the case-based abstraction hierarchy is the more complicated

of the two methods, requiring significantly more processing time and disk space, but this method is capable of performing non-linear classifications.

Once the user models have been created, when new articles need to be classified, the *global hill climbing* approach is invoked first. This approach involves the lookup of keywords and features matching a new article in the global hill climbing table, and then using the statistics associated with these features to compute a classification for the new article. Experimental results suggest that global hill climbing is useful as a quick, first-pass method that is simple to implement and relatively error free. However, if the global method returns an unknown classification, then the case-based reasoning module is invoked. This approach attempts to find case articles in memory that are similar to the new article by matching the concepts present in the new article with those in the case base. The classification of retrieved cases is then used to classify the new article. The case-based method may run in parallel with the global method although this was not implemented in INFOS. The case-based reasoning module may also be invoked separately in order to perform information retrieval from previously read documents. In addition to the case-based process, limited parsing of the input articles may also be performed to further refine the filtering or retrieval process.

Finally, a genetic algorithm is applied to a population of user models in order to explore the space of news articles. Genetic algorithms are based upon Darwinian survival of the fittest; user models compete with each other, and those which perform well survive and create offspring, while those which perform poorly die. Over many generations, the resulting user models become more and more “fit” and model the user’s interests more closely. By combining a variety of user models together, the space of news articles is explored while also converging upon a user’s interests.

1.6 Outline of Dissertation

Chapter 1 has provided an overview of this work and an introduction to the issues involved with Usenet news and information filtering. Chapter 2 explores these issues from a psychological perspective. In addition to describing psychological issues that users experience during information overload, an experiment was conducted to determine whether or not an information filter is necessary for Usenet news and to examine patterns of user behavior as news articles are read.

An overview of the architecture and algorithms implemented in INFOS is described in chapter 3. A detailed analysis of the global hill climbing method and experiments is given in chapter 4, the case-based reasoning method in chapter 5, and the genetic algorithm scheme in chapter 6. The technique of partial parsing via index patterns to improve the performance of the schemes discussed in chapters 4-6 is described in chapter 7.

A description of related work in information filtering is given in chapter 8. This chapter includes work in information retrieval as well as user modeling. Finally, the current status of INFOS and future work is presented in chapter 9, and concluding remarks about the entire work presented in chapter 10.

2. Psychological Motivation and Experiment In User Behavior

What is information overload? In Stevens' thesis (Stevens, 1992), information overload is defined from a psychological perspective. A common measure of information overload is that overload occurs when humans are presented with 10 or more items which fill short-term memory, or when a decision maker's capabilities cannot operate quickly enough to attend to all of the incoming information (Simon, 1981). Depending on the domain, information missed may be crucial (e.g., air traffic control) or minor (missing an interesting article). One of the results of information overload is boredom; Klapp (1986) reported that boredom is proportional to the amount of noise in the information, but the definition of noise varies with different users. Consequently, it is important for any filtering algorithm to conform to individual user preferences rather than pre-defined parameters or stereotypes.

When confronted with an overload of unfamiliar items, Thukral reports that people use a negative bias heuristic to select the items to examine in more detail (Thukral, 1983). This entails filtering out items based upon negative characteristics, rather than selecting items based upon positive characteristics. Once the number of items was reduced, positive attributes were considered. These human heuristics suggest that similar methods for a computer filtering system may result in similar performance. Moreover, both positive and negative heuristics should be considered.

Actual strategies suggested for humans to reduce information overload can be categorized as (Shick et. al., 1990) managing time more efficiently, fewer tasks to perform, having more time available, and expanding the size of the workforce. All of these strategies can be utilized through an intelligent filtering system; time will be managed more efficiently since users will have less information to deal with, resulting in more time

to concentrate on the selected articles. The filtering system itself can be viewed as an agent or worker to help the user discriminate different messages.

In the domain of news filtering, a study by Stadnyk and Kass identified five key categories readers used to discriminate among messages (Stadnyk & Kass, 1992). These categories include Domain concepts (semantics describing the subject of a message), Goals (user interests), Message types (message classes), Message characteristics (contextual information about messages), and Relations (relations between goals and domain concepts). Stadnyk and Kass propose rules that apply to these primary category types to aid in modeling important user features so as to improve information filtering. Their results imply that a filtering agent will require knowledge of the domain and the user for best results.

An additional psychological problem that relates to Usenet News is the "vocabulary" problem (Furnas et. al., 1987). This problem refers to the misconceptions based on ambiguous terms of different newsgroups. For example, a user may mistakenly search the newsgroup "comp.sources.unix" for source code that is really indexed under "comp.sources.x". Stevens attacks this problem by allowing users to define their own virtual newsgroups (Stevens, 1992). Another possible approach to be employed in this project is to have the filtering system suggest articles likely to be of interest from newsgroups the user may not be actively reading.

2.1 Examining User Behavior with Usenet News

The psychological issues surrounding information overload suggest that principles such as flexibility, positive heuristics, and negative heuristics should be integrated into a filtering system. However, although a number of systems have been created to filter Usenet news, an important basic question has remained unexamined: Is a filter even necessary in this domain? Studies have not been performed to determine whether or not

filters are necessary. The users' current form of browsing may already give adequate performance. How many articles are users currently reading that they would prefer not to read? Conversely, how many articles are users not reading that they would like to read? We conducted a study to answer these questions.

In this study, the classification of articles was compared when users browsed articles with a conventional news reader with situations when users were forced to read all articles. In a conventional news reader, users are given a list of articles in which the author and subject are displayed as shown in figure 3. As described previously, the subject of a message does not always accurately describe the content of a message. This experiment investigated whether or not this format for displaying messages provided sufficient information for users to pick messages of interest accurately.

The experimental results that follow indicate that the current system of browsing results in many messages that users do not read, but would be interested in reading. Furthermore, the results indicate that users often change their mind about whether they like or dislike a particular article. These results suggest that a news filter would be a great aid in finding articles likely to be of interest that are normally missed and that the accuracy of such a filter will be limited due to human inconsistencies.

2.2 Examining User Behavior - Experimental Method

The newsgroup selected for this study was the **ucd.life** newsgroup. This newsgroup was selected since all of the subjects in the study were UC Davis students and the newsgroup covers a variety of topics likely to be of interest to the general community. Most other newsgroups were too narrow to contain messages of interest to a general group of subjects. Furthermore, this newsgroup receives moderate traffic (approximately 50 messages a day) so that filtering may be useful. The subject matter of this newsgroup

varied from Want-Ad postings to discussing crime in Davis. A selection of topics from the articles used for this experiment are:

Any bad experience with J str. Apts?
Art 111 home page
Bicycle geeks
Bunbun's reality revisions
Furniture for sale
Unabomber in Davis
Davis Police Department
Proper English

144 sequentially posted messages from the newsgroup were selected for the study. These messages were sorted into threads and displayed to the user in the standard news reader fashion, giving the author and subject, as shown in figure 3. Users were first instructed to browse the articles as they normally would, and read those articles that looked interesting. After a user read an article, the system asked the user to classify the article as being **accepted** if she was glad she read the article and found it of interest, **rejected** if she really did not want to read the article, or **unknown** if she is unsure or ambivalent. In this manner, all of the articles the user decided to read during browsing were assigned a classification of accepted, rejected, or unknown.

After the browsing phase was complete and the subjects were satisfied that they had read all the messages they felt would be of interest, the subjects were instructed to read all 144 messages. For each message, users gave a classification of accepted, rejected, or unknown. If the existing methods for displaying articles is sufficient and no filtering is necessary, then the message classifications during the browsing phase should closely match the message classifications from when all messages are read.

A total of 14 unpaid volunteer subjects participated in this study. All subjects were UC Davis students, 2 of them graduate and 12 of them undergraduate students. All subjects were familiar with existing news readers and had read the ucd.life newsgroup in the past. With the exception of the graduate students, the subjects were naive about the purposes of the experiment.

2.3 Examining User Behavior - Results

The results showing the classifications of messages for each test subject and their totals are shown in table 1. Articles that were not read during the browsing phase were classified as unknown, along with articles the user read and classified as unknown. In addition to tallying the totals, the flip-flops were also counted. Flip-flops count the number of messages read during both the browsing phase and the all-messages phase, but which were classified differently by the subject.

	Browse Accept	Browse Reject	Browse Unknown	Total Accept	Total Reject	Total Unknown	Total Flip-Flops	Flip-Flops - to +	Flip Flops + to -
Subj 1	6	9	129	81	30	33	8	7	1
Subj 2	5	6	133	45	3	96	8	5	3
Subj 3	32	6	106	42	82	20	3	2	1
Subj 4	4	1	139	42	82	20	3	1	2
Subj 5	24	9	111	80	39	25	11	5	6
Subj 6	33	23	88	58	41	45	11	7	4
Subj 7	7	5	132	48	66	30	7	4	3
Subj 8	14	9	121	36	107	1	8	2	6
Subj 9	33	18	93	40	43	61	13	7	6
Subj 10	21	2	121	8	7	129	19	1	18
Subj 11	13	6	125	73	29	42	8	6	2
Subj 12	17	4	123	68	35	41	3	3	0
Subj 13	2	8	134	9	131	4	1	1	0
Subj 14	1	3	140	67	42	35	1	1	0
Total	**212**	**109**	**1695**	**697**	**737**	**582**	**104**	**52**	**52**

Table 1: Classification results for subjects browsing messages and reading all messages. During browsing, messages classified as “unknown” are grouped with messages not read. The Flip-Flops column indicates the number of messages whose classification was changed by the subject from the browsing phase to the reading-all phase. The - to + values indicate flip-flops from reject to accept, while the + to - values indicate flip-flops from accept to reject.

321 messages were classified as accepted or rejected by the collective subjects during the browsing phase. Of these 321 messages, 104 were classified differently when the subjects were forced to read all messages, resulting in a flip-flop rate of 32%. The percentage of messages accepted during browsing is 11%, while the percentage of

messages rejected during browsing is 5% and the percentage of unknown messages is 84%. 2016 messages were read during the read-all phase. 37% of these messages were marked as accepted, 41% as rejected, and 22% as unknown.

2.4 Examining User Behavior - Discussion on Patterns of Behavior

One of the results shown by this study is the different reading patterns among subjects. Some subjects, such as subject 14, read only a few messages during browsing, while others read up to 50. Subject 10 displayed erratic behavior, flip-flopping on almost every message read during browsing. A few subjects rejected almost all messages, while others accepted almost all messages. The end result is that every user is different and that users are often inconsistent in their behavior. For a filter to accommodate all of these behavior patterns, such a system must be able to adapt to individual preferences rather than conform to any norm or user stereotypes.

Despite the wide range of differences in reading patterns, there are a number of distinct modes of operation: (1) browsing in general, (2) searching for specific information, or (3) simply reading every message. In browsing, users scan through the list of messages looking for messages of interest. Typically the reader is not looking for anything in particular, but whatever topics may be of interest. In this experiment, the test subjects were thrust into this category by nature of the experiment. However, one subject reported that after browsing and reading an interesting topic he searched the remaining messages for similar topics. In search mode, users have a specific agenda that they are searching for. For example, if a user is interested in buying a SCSI CD-ROM, she may scan through newsgroups looking specifically topics related to CD-ROM's, SCSI, or Mass Storage. Finally, some users simply read all messages. This is often the case for low-volume newsgroups that only receive 10-20 messages a day, but it is not possible n for high-volume newsgroups.

These different modes of reading imply that a filtering system must not only be flexible enough to adapt to individual user preferences in message topics, but also to the user's current mode. In browsing mode, the system must identify messages likely to be of interest based upon previous user feedback. In search mode, the system must allow the user to input search queries to narrow the filtering process. Finally, for users who read every single message in a newsgroup, there is no need for a filter and any filtering system should be turned off.

2.5 Examining User Behavior - Discussion on Finding Messages of Interest

During browsing, the subjects indicated that they were interested in only 212 messages. However, when the subjects read all messages, they indicated that they were actually interested in many more messages. A total of 697 were accepted, almost three times the number read during browsing. Subject 14 is an excellent example of this phenomenon. During browsing, subject 14 accepted only one article, but later was really interested in 67 articles.

One explanation for these results is that displaying messages by author and subject alone do not provide enough information to allow users to pick accurately the messages they would like to read. In interviews with the subjects after the experiment was conducted, 12 indicated that they found most of the articles to have a different content than originally expected from reading the subject header alone. Another explanation for the higher acceptance is that increased reading resulted in increased interest. Four of the subjects reported that they "started to get into it" after they started to read more messages. In other words, reading some messages generated additional interest in other messages resulting in more messages classified as accepted.

The types of flip-flops made also supports the theory that users' interest piqued after they began to read more messages. Although the flip-flops are evenly divided (52

from accept to reject, and 52 from reject to accept), the totals are biased by one single individual. Subject 10 flip-flopped on almost every message initially read, and all except one of the 19 flip-flops are from accept to reject. If data from this erratic subject is eliminated, then 51 (60%) of the flip-flops are from reject to accept, and 34 (40%) of the flip-flops from accept to reject. Consequently, more subjects tended to accept the articles they previously rejected.

To increase the chances that "missed" messages of interest are read, two direct approaches may be used. First, an intelligent filter could identify those messages likely to be of interest and alert the user. This works only as long as the user trusts the system and what messages the filtering system recommends. Second, the interface used for browsing could be improved to include content from the body of each message to give the user a better indication of what the message is about. This should allow readers to make a more informed choice of which message to read.

On the opposite spectrum of finding articles of interest is rejecting articles not of interest. With over 36% of the messages classified as rejected, this comprises a majority of the three classifications (35% accepted, 29% unknown). This volume of rejected messages indicates that the capability to recognize these articles will certainly aid the reader in selecting relevant articles. Note that since this is the most prevalent classification, the classification problem is non-trivial. Randomly selecting a classification with equal probability would result in only a 33% correct classification. Similarly, always selecting the most prevalent class of "rejected" would result in 36% correct, but a high error of 64%. However, this applies only to the subjects as a whole. Individually, applying the trivial classification of all rejected works well for a few users, such as subjects 8 and 13, who rejected almost all articles.

2.6 Inconsistency of User Interests

One of the unexpected results of this study was the high number of flip-flops; subjects who classified a message one way during the browse phase, then later classified the message differently when all messages were read. Out of the 321 messages classified during browsing, 32% of them (104 messages) were changed during the read-all phase. One of the reasons for this change is the increased user interest resulting from reading more articles, as described previously. Furthermore, subjects typically read very few messages during the browsing phase. This limited exposure to articles is not enough to gauge accurately what threads of conversations are about. Additionally, some subjects had a narrow threshold between acceptance and rejection. Depending upon the context, or even the mood of the reader, articles could be classified either way.

These flip-flops raise an issue about the maximum possible performance a filtering system can achieve. With 32% of the classifications changing, a very large error will result due to the fickleness of the readers. Many of these flip-flops stem from the limited number of articles initially read by the user. In an ideal setting, a user would only browse the messages she is interested in reading, and based upon these the system will filter future articles. However, this study has shown that users do not read enough browsed articles to build up a model of user interests accurately. Some of this error can be reduced by forcing users to read more messages, providing additional context for the articles. Nevertheless, in the end, any filtering system is subject to the whims and inconsistencies of the human user, making 100% accuracy virtually impossible to achieve.

2.7 Chapter Summary

This chapter has explored some of the psychological issues regarding information overload and presented results of an experiment that examined user behavior with Usenet news. Highlights of the experiment include:

- Subjects browsed articles through a standard news reader, and classified browsed articles as interesting, disinteresting, or ambivalent. The subjects later read and classified every article.

- When browsing, subjects missed many articles they were interested in reading. Additionally, subjects read many articles they were not interested in reading. An information filter may help a user select articles of interest while also filtering articles of disinterest.

- Readers are inconsistent and change their minds about whether or not they are interested in the same article. This inconsistency limits the performance that a filtering system may achieve and suggests the need for flexible filters and user models.

3. Overview of INFOS

Based upon the results of the previous experiment and work performed by other researchers, INFOS has been designed to address issues dealing with the users of an information filtering system, the structure of newsgroups, and the algorithms of an information filtering system. The overall goal of INFOS was to create a Usenet news filter with better performance than traditional keyword based systems through the incorporation of a semantic knowledge base and a wide range of extracted features. Furthermore, INFOS is capable of learning from user feedback alone, as opposed to some systems that require a knowledge engineer to create or maintain a knowledge base. The effectiveness of INFOS was then examined via user-testing and comparisons with traditional information retrieval techniques such as tf-idf.

3.1 User's Perspective of INFOS

From a user's perspective, users are likely to read messages in one of four possible modes: (1) reading all articles, (2) browsing a large number of articles, (3) searching for a specific topic among new articles, or (4) searching for a specific topic among previously read articles. To support the first mode, INFOS has the simple capability to be turned off or on. If all articles are being read by the user, then filtering is an unnecessary step.

To support the second mode of browsing, INFOS performs learning and filtering. INFOS automatically builds up a profile of user interests based upon active feedback, and then uses this profile to predict whether or not new articles will be of interest. In this mode, a user first selects a newsgroup to read and browses through articles. After each article has been read, INFOS asks the user to rate the article as Accepted if the user liked

the article, Rejected if the user dislikes the article, and Unknown if the user is unsure. This is a form of active feedback, since the user is required to rate each article specifically, but does require minimal effort and provide direct feedback. The feedback is used to create a user profile. Most systems require users to specify their own profiles, but many users are unwilling to spend the time or effort to do this on their own (Stevens, 1992). In contrast, by rating each article, INFOS gets accurate data on user interests in a non-intrusive fashion. An even more non-intrusive method is to use passive feedback. In passive techniques, the system gets no direct feedback from the user, but instead watches for cues such as read time or messages that are saved or replied to. Messages that are replied to are presumably messages of interest. While convenient, the user profile data retrieved from these methods is not as rich as an active approach.

To create a good profile, the experiment in chapter 2 suggests that users need to read more messages than just the messages they would normally browse. INFOS allows users to browse messages freely, but a better profile is created if users are forced to read a randomly selected subset of the new articles. Experiments were conducted using a variable number of randomly selected messages; these results are shown in chapter 4.

After a profile has been created, when new messages are read INFOS will classify the unread messages using the user profile and the filtering algorithm. The new unread articles are sorted into three categories: Suggested, Unknown, and Not Suggested. The articles INFOS suggests are displayed in threads with the Suggested articles listed first, the Unknown second, and then the Not Suggested articles last. In this manner, readers can quickly find the suggested articles, but can also see what articles the system believes the user is not interested in. In this manner, users have the opportunity to see how INFOS is classifying articles, and also has the opportunity to change the user profile if desired.

To give users a better idea of why INFOS is classifying articles the way it is, highly weighted keywords that contribute to the classification are also displayed on the

browsing screen. This allows users to get an idea of what factors contribute to the classification of an article, and if the classification is incorrect, to modify the user profile accordingly. Giving the user this feedback is an important user interface tool that has been lacking in other information filtering systems. By having feedback displayed to the user, the user will feel more comfortable with the filtering system and the system gains a more accurate model of user interests. While this feedback can be very useful, users can completely separate themselves from the filtering process if desired. No knowledge of how the filtering algorithms work is necessary to operate the system; all of these processed operate transparently without user intervention. However, by allowing users to get feedback about their profiles requires that the user profiles be simple to understand and modify. If they are not easy to understand, the average user will not take the time to edit their own profiles (Stevens, 1992).

The provision of keywords in the browsing screen can sometimes provide insight into the content of a message, but also the subject, author, or other terms are displayed. As postulated in chapter 2, one of the problems with conventional browsers is that no content from the body of the article is displayed and often the subject heading is not related to the text itself. INFOS partly addresses this problem by also displaying the first full line of new text from the body of each message in the browser screen. While the first line is not always relevant, this at least allows users to have a glimpse into the body of a message.

A listing of messages as shown by INFOS while browsing is displayed in figure 4. Messages are sorted with those suggested (+) at top, unknown (?) next, and not suggested (-) at bottom. Author, subject, contributing factors used to make the classification, and the first line of the body of each article are displayed to help the user browse intelligently. In figure 6, INFOS believes the reader is interested in messages from Travis Higgins regarding concerts, and from Michael Duran regarding school

advice. Reasons for the articles given a negative rating include collaborative reviews from Zhou, usernames such as dtwitko, and words such as sale or wanted.

242. +Travis Higgins	Central Park Concert needs your help!
	Key Factors: central concert
	Tour For Peace Free Concert is coming...
243. +Travis Higgins	Re: Central Park Concert needs your help!
	Key Factors: central concert
	The intent of the Tour for Peace is...
235. +Michael Duran	I need grad school advice
	Key Factors: school advice
	After four years at Davis and thinking...
213. ?Andrew	Any bad experience with J str. Apts?
	Key Factors: experience problem
	Has anybody had bad experiences with...
215. ?Davis Police Department	Re: Any bad experience with J str. Apts?
	Key Factors: experience dpd@whe
	The City of Davis Community Mediation...
200. ?Kim Nguyen	Art111 Home page
	Key Factors: zhou_reject ez01840
	Announcing the Art111/Advanced Photo...
222. ?Chimp	Proper English
	Key Factors: proper english
	I reluctantly offer my support to Jim...
221. ?Rudeboy	SIDEMEAT Reminder
	Key Factors: ez01898 reminder
	Just a reminder: The First Annual...
233. ?Blaise Camp	Re: Today's the Day
	Key Factors: ez03051 hope
	Well, it got to a late start, but...
214. -Henry Tesluk	Re: Any bad experience with J str. Apts?
	Key Factors: experice szht@ucdavis.edu
	Andrew Oleinikov wrote...
205. -irie	drummer wanted
	Key Factors: wanted irie
	Now auditioning local drummers for...
207. -David T. Witkowski	Re: drummer wanted
	Key Factors: dtwitko zhou_reject
	How many times are you going to post this?
217. -Sharkmn435	Furniture for sale
	Key Factors: zhou_reject sale
	A couch andlove seat, nice blue color...
236. -Davis Police Department	Have you ever wanted to know...
	Key Factors: wanted dpd@whe
	Hello, I am on-line representing...
237. -kari orkney	Re: Have you ever wanted to know...
	Key Factors: wanted zhou_reject
	Isn't Davis, CA the home of the famous...

Arrows Move,(P)rior Screen,(N)ext Screen,(Q)uit,Return reads

Figure 6: INFOS browsing screen.
Messages are sorted with those suggested (+) at top, unknown (?) next, and not suggested (-) at bottom. Author, subject, contributing factors used to make the classification, and the first line of the body of each article are displayed to help the user browse intelligently.

To support the third mode of searching for a specific topic among new messages, INFOS allows users to define their own interest profiles and to turn learning off but leave filtering on. When search criteria have already been defined, learning is not necessary - performing the learning algorithm may even result in corrupting the original search profile if the reader begins to stray from the original search topic. Defining search criteria is achieved by modifying the user profile. This is described further in chapter 4.

To support the fourth mode of searching for a specific topic among previously read messages, INFOS allows users to enter keywords to search for relevant documents. The search engine used for information retrieval is based upon the same engine used for information filtering. Through a semantic memory hierarchy (see figure 18 in section 5.2), retrieval is *conceptual* in addition to keyword based. This allows articles to be retrieved that are similar in concept to the search query. For example, a search query regarding motorcycles will not only match documents about motorcycles, but to a lesser extent match articles regarding bicycles. To an even lesser extent, this query will match articles regarding automobiles or other transportation vehicles. In contrast, a purely keyword based information retrieval system is limited to retrieving only those articles that contain the exact same keyword.

In addition to supporting these modes of operation along with article filtering, INFOS also supports a number of standard news reader functions:

- Threading of news articles
- Posting of news articles
- Mailing / Forwarding of news articles
- Article reply via Post or E-Mail
- Quote replied document in reply
- Partial auto-extraction of UUencoded articles

3.2 Newsgroup Structure - Vocabulary Problem

Words often mean different things to different people; this is the basis of the vocabulary problem (Furnas, 1987). In addition to people's use of like terms differently, words are also ambiguous. As a simple example, the word "mac" in a computer newsgroup is probably referring to a macintosh computer, while the word "mac" in a fast-food newsgroup is probably referring to a big mac hamburger. The additional information that allows a reader to determine which meaning is appropriate is the context in which the word appears; not only the context of the other words in the document, but also the context of which newsgroup the message is being posted to. To use this contextual information, INFOS keeps a separate user profile for each individual newsgroup. In this manner, identical words that are used differently in other newsgroup contexts are evaluated separately without interference. However, in many cases terms from one newsgroup is applicable in another. This information can be used to help explore the problem space, as in Sheth's NewT system (1994).

3.3 Raw Features Used for Information Filtering

Before information filtering can take place, features must be extracted from the textual articles in order to build the user profile. In INFOS, four types of raw features are extracted: (1) the author of an article, (2) tokens from the subject of an article, (3) tokens from the body of an article, and (4) collaborative information. Note that these features include almost all relevant data for an article, while some systems only extract the authors and subject to simplify the filtering process. Additionally, these features include only raw features; during the case-based reasoning phase, INFOS further refines the tokens from the subject and body into semantic indices.

Author extraction parses the message header and removes the author's e-mail address for use as a feature. This feature allows indirect filtering since the author of a post does not necessarily indicate the content of the article. However, individual authors may tend to post similar material, and many readers often have favorite authors. Additionally, this allows users to track postings from a particular user. For example, this can be used to see what type of work a particular researcher has been investigating and discussing over Usenet.

Subject and text body extraction parses the actual words from the subject line of the article and the body itself. In INFOS, all punctuation is removed (with the exception of periods to denote the end of sentences). These features support direct content-based filtering since they determine what an article is about. Since the subject text is intended to indicate the content of the body, the subject's features are weighted higher than the body's features.

Social or collaborative feature extraction is a relatively new subject in artificial intelligence, and research has just begun in such diverse areas as music reviews, WWW selection, and news filtering (Goldberg et al. 1992; Resnick et al., 1994; Shardanand, 1994). Collaborative features are simply the recommendations or reviews that other users have reported about a particular item of interest. This is similar to listening to movie reviews from movie critics. For example, based upon Siskel's opinion, a viewer may or may not decide to watch a particular film. If the viewer has a strong rapport with Siskel, or has enjoyed Siskel's recommendations in the past, then the viewer will most likely enjoy a new movie that Siskel recommends. In this project, news articles are the "movies" and early readers of those articles are the critics. The ratings that readers give to articles they read are used as features for future readers of that article. These features determine if there are correlations between the articles one reader is interested in and the articles other readers are interested in. Although collaborative features are indirect since they do not reference the actual content of the article, this type of feature can be very

powerful since it incorporates the collective semantic knowledge of many other users. Moreover, collaborative features can be used in virtually any setting (music, text, movies, graphics, etc.).

3.4 Architecture of the Information Filtering Engine

To support the goals established for INFOS, the system must not only be flexible enough to accommodate usability, but also powerful enough to filter documents accurately. One of the greatest challenges is the design of the filtering engine itself so that these goals can be achieved with efficiency and precision. A semantic knowledge base is one method to gain increased flexibility and precision over a keyword system. The dilemma is, how can a system be designed so that conceptual retrieval is possible yet without the costly time or expertise necessary to create a knowledge base? In the domain of Usenet news, articles cover a very wide range of topics, messages are unstructured, and the vocabulary is constantly changing. The knowledge required for a symbolic knowledge based system is enormous, and it is difficult to manually create the lexicon, rules of syntax, and other knowledge constructs necessary for in-depth language processing. These constraints typically restrict knowledge-based systems to limited domains, and they have frequently been criticized for failing to "scale-up" to real-world applications (Schank, 1991). The approach taken in INFOS combines limited real-world knowledge with statistical methods to achieve the scalability of keyword systems while retaining some of the power of knowledge-based systems. In addition to combining the benefits of scalability, gains are also made in filtering speed and accuracy. Keyword based filtering is faster than knowledge based filtering, and it is also capable of handling new words missing from the knowledge base's lexicon. On the other hand, knowledge based filtering supports conceptual information filtering and retrieval. By combining both approaches, the benefits of each can be achieved in a single system.

The algorithms and knowledge structures combined in INFOS consist of a hybrid induction and case-based reasoning system augmented by the hypernym ISA structures found in the WordNet system (Miller, 1995). WordNet is similar to an electronic dictionary, such as Webster's Seventh New Collegiate Dictionary (Peterson, 1982) or Longman's Dictionary of Contemporary English (Alshawi, 1987), but WordNet defines a word in terms of its hierarchical meaning rather than in terms of other words. The hierarchical knowledge base allows concepts rather than individual words to be compared to each other.

Filtering of articles is first performed by a simple keyword approach we have named Global Hill Climbing. This approach is a slightly simpler version of tf-idf, and has been shown to work fairly well in most cases. Global Hill Climbing consists of matching a global table of extracted features and their previous frequencies of acceptance or rejection with an incoming document to classify the new document as being of interest or not of interest. The method is simple and quick, has been shown to have a very low error rate in experimental tests, and is easier for users to modify their profiles than tf-idf. However, the scheme does result in a fairly large Unknown classification rate of approximately 40% (Mock & Vemuri, 1994).

When the Global Hill Climbing approach fails, a more robust but potentially time-consuming Case-Based Reasoning (CBR) approach may succeed - in particular, because it performs conceptual information filtering that may retrieve articles the keyword scheme cannot. The principle behind case-based reasoning (Schank, 1982) is that inductive learning is accomplished through the memorization of individual experiences, or *cases.* These cases are simply experiences of the learner that have been remembered. When new situations are exposed to the learner, the learner is reminded of previous cases which are similar to the new situation. The actions and events that took place in the old case are used to help understand how to deal with the new case. The sub-field of case-based reasoning has been successfully applied to many applications,

including such diverse areas as help-desk systems, failure analysis, cooking, and law (Kolodner, 1993).

CBR has been selected for use in INFOS since it has been successful in the past, provides a natural framework for information retrieval as well as information filtering, and provides a framework upon which a knowledge-based representation can be built. In INFOS, the case-based reasoning component treats previously read documents as cases to help understand new documents. The previously read documents are indexed by both conceptual and keyword indices. These indices are extracted statistically and stored conceptually using the WordNet hypernym hierarchy. Articles that match these indices and the user's previous interest in those articles indicate the classification of new articles. In this manner, articles are tracked independently unlike the Global Hill Climbing method. The case-based reasoning scheme also facilitates information retrieval of previously read articles along with information filtering. Moreover, the retrieved article cases from the case-based reasoning method provide a justification for how a new article is classified.

An additional benefit of the case-based reasoning scheme is that the engine is directly applicable for case-based document retrieval. This provides a convenient mechanism to support conceptual search among previously read documents. The identical process applies to document retrieval as it does to document filtering, except instead of comparing a new article to previously read articles, a user-supplied query is compared to previously read articles through both keywords and the conceptual indices. The candidate documents are then simply displayed to the user rather than used to compute a classification.

A final algorithmic process to be applied to the user models is the genetic algorithm (Holland, 1975). In the genetic algorithm, a population of solutions or *individuals* are randomly generated to solve a problem. Typically these individuals are represented by strings of bits, where each bit represents a different aspect of the solution.

For example, a numerical solution could be represented by its binary value. Since these solutions are randomly generated, they will initially be very poor solutions. However, some solutions will be better than others, and if enough solutions are generated to "cover" the solution space, then portions of different individuals can be combined to form an optimal solution. The genetic algorithm (GA) approaches this task through the process of natural selection. In nature, Darwinian evolution stipulates that the fittest species for their environment will survive, while the weak will die. Those animals that survive pass their surviving characteristics on to their offspring through genetic crossover, resulting in an even stronger population in the next generation. The same process is applied to the computational GA. The individual solutions in the population are subjected to a *fitness* test to determine how well each individual approximates the desired solution. Those individuals that do well survive and are selected for *crossover* while those individuals that are poor solutions die and are thrown out of the genetic pool. The surviving individuals form a new population by taking two random parents and crossing portions of their bit patterns to create a new child. In this fashion, the child will inherit the surviving attributes of both its parents, possibly resulting in an even fitter individual. Finally, random mutation is sometimes applied to the offspring to introduce a random element to prevent the gene pool from stagnating. Eventually through many generations, the population becomes better and better at approximating the true solution. Moreover, since the process contains a random element, there is a smaller chance that the solution will converge on a local minima than classical techniques such as hill climbing (Goldberg, 1989).

For information filtering, genetic algorithms are useful when applied to a population of user models. These models then converge upon a solution that better models the user's interests. In NewT, Sheth has shown that genetic operators such as crossover and mutation aid the user in exploring other areas of the news space that the user is interested in, but does not normally explore (Sheth, 1994). Similar results have been recreated in INFOS's preliminary work (Mock & Vemuri, 1994), which also

indicates that genetic algorithms are useful for exploring other areas of the solution space. However, in this work, the GA was found to be better at exploration than at narrowing down user interests.

3.5 Summary of INFOS Architecture

A summary of the information filtering classification process is depicted in figure 7. To classify new articles, the global hill climbing approach is used first. The experimental results discussed in chapter 4 indicate that the global hill climbing approach has a very low error rate but a fairly large proportion of unknown classifications. These results also suggest that global hill climbing is useful as a quick, first-pass method that is simple to implement and relatively error free. Consequently, if the global method returns an unknown classification, then the case-based reasoning module is invoked. The case-based method may be run in parallel with the global method although this was not implemented in INFOS. The case-based reasoning module may also be invoked separately in order to perform information retrieval among previously read documents. Details of both methods are described in chapters 4 and 5.

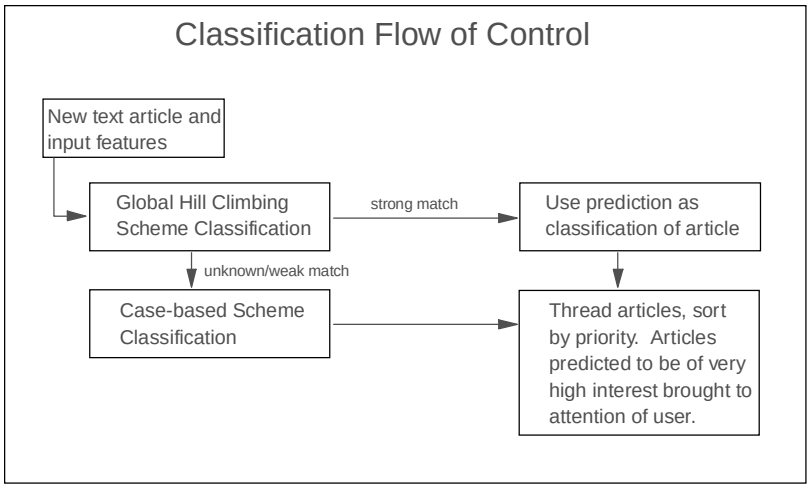


Figure 7: Classification Flowchart

After new messages are read, the process of updating memory and the user model is depicted in figure 8. First, the global hill climbing table is updated with features from articles that were read and rated. These features include the author, words from the subject header, words from the text body, and collaborative reviews from other users. Next, conceptual topics from each article are identified and used as indices into the case-based reasoning component of the system. This is the more complicated of the two classification algorithms and requires significantly more processing time and disk space, but is capable of performing non-linear classifications. Details are also described in chapter 5.

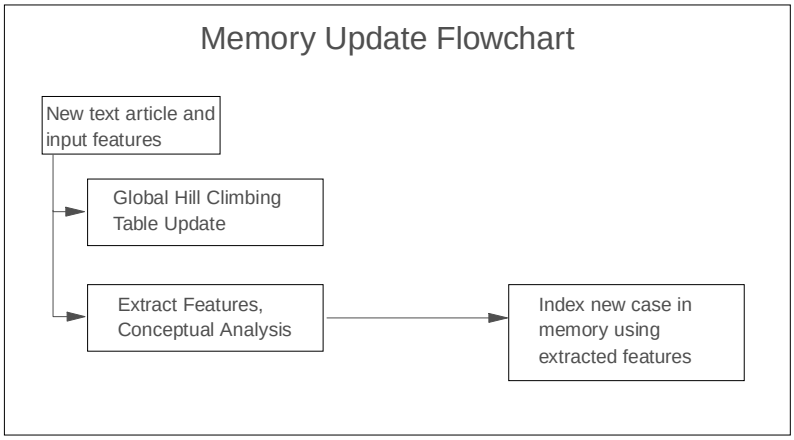


Figure 8: Memory Update Flowchart

From the user's perspective, the following features have been implemented to improve the browsing and user modeling processes:

- A means of showing the user what articles are being filtered. Chapter 2 has shown that users interests vary tremendously. If articles classified as rejected are simply discarded then messages of interest are likely to be lost. INFOS sorts articles by interest category so that users are still able to view messages not recommended by the system, allaying any fears users may have of missing out on interesting material.

- A means of showing the user why the system is making its classifications. In INFOS, this is accomplished by displaying the largest contributing features towards the classification. In addition to providing additional information regarding the content of each message to the user, by displaying the contributing features the user is able to see how the user profile is currently defined. If desired, the user may then modify the profile to reflect different interests. Profile modification requires that the user profile be easy to understand and modify so that it can be manipulated by a typical end user. Moreover, by making the profile easily modifiable, users will be able to create profiles that conform to specific search queries.

- A provision for information retrieval in addition to information filtering. In this mode, users can retrieve previously read articles by inputting search queries. Through the same CBR and keyword based engine used for filtering, queries are conceptually analyzed and those documents that contain a similar conceptual content are retrieved. This type of retrieval is made possible through the CBR filtering engine.

3.6 Chapter Summary

This chapter has provided an overview of the INFOS system from both the user's perspective and INFO's internal architecture. The highlights of INFOS's architecture and interface include:

- Automatic creation of a user model based upon user feedback provided after the user reads each article.

- Support for editing the user model easily, and a mechanism to communicate back to the user how and why INFOS makes its classifications.

- A hybrid classification scheme incorporating a global hill climbing method that uses keyword features and a case-based reasoning method that uses knowledge-based features. Between user sessions, a genetic algorithm is employed to explore the space of news articles and further refine the user model.

- Features used for classification within the global hill climbing scheme include keywords from the body of the news article, keywords from the subject, the author, and collaborative reviews from other users. Features used for classification within the case-based reasoning scheme include WordNet based concepts of keywords from the subject and body of the news article.

4. Global Hill Climbing Filtering Algorithm

The work reported here builds upon an initial prototype studied by Mock and Vemuri (1994). The prototype consisted of the Global Hill Climbing scheme and a genetic scheme. These schemes remain in the current version of INFOS, but they have been augmented with the case-based reasoning scheme described earlier. In this chapter, the global hill climbing algorithm is described in detail along with the experimental results when the algorithm is applied to the filtering task.

4.1 Usenet Data

The data used for the filtering experiments consisted of the articles from the **ucd.life** newsgroup selected for the experiment in chapter 2. These articles were selected for the same reasons as the previous experiment; there are a large number of articles, and the articles cover a variety of general topics with appeal to a mass audience. After articles were read, the author, subject tokens, body tokens, and collaborative review data were all extracted as features of the articles. The tokens were first passed through a *stop list*, but not through a *stemmer*. A stemmer attempts to strip away word prefixes or suffixes to find the word root, so that words such as “finding” or “finds” are reduced to “find” for comparison purposes. A stop list is a list containing common words such as “the” that have no predictive value. These words are thrown out entirely. The accuracy of available stemmers is limited and there is a question of whether or not they actually improve performance (Yang, 1994).

In addition to passing the articles through a stop list, extraneous headers from the articles are stripped, binary encoded files thrown out, and quoted material removed. A sample news header is shown in figure 9. The only features currently used from the

header as features in INFOS are the subject, author, and indirectly, the newsgroup. The rest of the information is easily discarded since the header is formatted in a standard manner.

```
From: tswillia@menudo.engr.ucdavis.edu (Bunny Bunbun)
Newsgroups: ucd.general,ucd.life
Subject: Re: Happy May Day
Followup-To: ucd.general,ucd.life
Date: 4 May 1995 08:04:46 GMT
Organization: College of Engineering - University of California - Davis
Lines: 64
Message-ID: <3oa1qu$cer@mark.ucdavis.edu>
References: <dtwikowski-0305952302120001@dcn73.ucdavis.edu>
NNTP-Posting-Host: menudo.engr.ucdavis.edu
X-Newsreader: TIN [version 1.2 PL2]
Xref: news.ucdavis.edu ucd.general:1325 ucd.life:3855
```

Figure 9: Usenet Header Information

Binary encoded files are also completely discarded. Binary encoded files are binary files that have been converted into text form so that they may be distributed through newsgroups or mail. Figure 10 depicts a portion of a UUencoded file. Since there are no conceptual features to extract, these files are currently not processed by INFOS. Some typical binary encoded formats include UUencode, MIME, Binscii, and BinHex. While the UUencode and MIME formats are detectable through identifying keywords in the header of the encoded file, other methods do not have identifying headers. These formats are detected by checking if the frequency of symbols in the first 10 lines is dramatically different from the frequency of symbols in English (over 10% symbols in each line), and also checking to see if the first 10 lines are the same exact length. If both of these criteria are satisfied, the article is marked as being an encoded file and is discarded. While these heuristics are not perfect, they were tested upon 100 articles randomly selected from the alt.binaries.pictures newsgroup containing primarily encoded pictures, and 100 articles from the ucd.life newsgroup containing completely text. No encoded

binary files were classified as containing text, and no textual articles were classified as being encoded binary files.

```
begin 644 binaryfile
M1TE&.#=A@ +@ 8,      (    "  ("     @(  @ " @(" @,# P/\   #_
M /__    __\ _P#_____RP     @ +@ 0 $_E ]M:2ZLTY,+=>7!7Z>6$YH
MF&Y6ZK36NRHR7+OP=,<YW>^ZGW &]!&'-B3NR$PVE\XH=,I[5J57:M"ZQ7:U
M1F\87"PKL^AO>KPV<]GJ.%P%XZ1(K,RID^?31R9]@"&$A8:'B(F*BXR-B R^&
MD(62E"&5%Y<^F):<FYZ:H)FBG"^5I:2A/J:IIZVJKZZQL+.RM;2 WMKFXN[J]
```

Figure 10: Portion of a UUencoded file

The final pre-filtering process involves discarding quoted material. Often when users reply to an article, they will include text from the original post to provide context for their comments. A sample is shown in figure 11. In INFOS, the quoted material is thrown out since the goal of the system is to classify the content of the existing article, not the previous articles that have been repeated in the existing article.

> In a previous article, Khristy wrote:
> > I was walking through Toys-R-Us the other day and came across some Floam.
> > Floam is this squishy styrofoam stuff that you can mold kind of like clay, but it is
> > more gooey. Does anyone know what this is made out of?
>
> Khristy, I'm not sure what floam is made out of, but I know that you can make a very similar thing by combining borax with elmer's glue, and then adding styrofoam. It's a lot of fun for the kids too.
>
> > Thanks, Khristy.
>
> You're welcome, Bobo.

Figure 11 : Quoted Material in Posted Articles

While most news readers use the “>” symbol to indicate which material is being quoted, readers are inconsistent and use a variety of schemes. Some readers use other punctuation symbols such as “:” or a “!”. A few users simply indent the quoted text, or preface quoted material by the author’s name, as in “khristy>”. Detecting all of these methods is an extremely difficult task requiring a large number of cognitive processes to

determine what material constitutes a quote. For example, to determine the difference between indented text that is quoted and indented text that is original requires knowledge of the content of previous articles, and a comparison of those previous articles with the current article. Identical material can then be discarded. Since these cases are rare, the heuristic employed in INFOS to detect quoted material is:

- Strip out any line beginning with ">" as being a quoted line
- Strip out lines where the first character is non-alphanumeric and at least 2 consecutive lines contain the identical character

This simple scheme was tested on 100 articles from the ucd.life newsgroup. With the exception of users who quoted material using indented text (2 articles out of the 100), all other quotes were successfully filtered, excluding signature files. Signature files were not filtered since they may potentially contain useful features, such as the author's name, location, or current interests.

4.2 Global Hill Climbing Background - A Simple, Keyword Scheme

One of the requirements for the user model is that it must be very simple for users to modify and understand; if the model is too difficult to manipulate, the average user will never use it (Stevens, 1992). In addition to simplicity, the model must also provide good performance. One of the models implemented in INFOS is a combination of the tf-idf method and Bayesian induction. Both of these methods operate upon keywords and features. A keyword/feature based system was initially selected for the user model since it is easy to perform computationally and also easy for users to understand.

4.2.1 Term Frequency - Inverse Document Frequency

A very closely related area to information filtering is information retrieval; while information filtering may be viewed as a simpler problem than information retrieval, both share essentially the same problem: extracting features from a document and using those features to index the document for future retrieval, or to generalize some type of class for the document. In the information retrieval community, one popular form of document indexing is term-frequency indexing coupled with the inverse-document frequency, which is referred to as *tf-idf* (Salton, 1991).

Tf-idf assumes that the frequency of occurrence of a term or keyword within a document is an indicator of how relevant that term is. However, if a term or keyword appears in many documents, then its predictive power is weak. For example, a common word such as "the" appears many times in one document, but appears in so many documents that it is a useless term that provides almost no information. These two terms may be combined by multiplying the term-frequency (*tf*) by 1/document-frequency (*idf*) to obtain a metric of relevancy for each term. By combining terms from a document to form a vector, queries can undergo a similar process and the document vector closest to the query vector is retrieved as the best match.

To express this process mathematically, the weight of a term *t* with respect to document *i* is described by:

$$weight(t)_i = tf(t)_i \times idf(t)$$

Since the tf term is a better predictor of a terms value than how common the term is, often the inverse document frequency is scaled to de-emphasize large weights by taking the logarithm of the frequency term *t* appears in all documents:

$$idf(t) = \log\left(\frac{1}{f(t)}\right)$$

Note that words are typically checked against a stop list. These words are ignored. With a good stop list, the words with a low idf value will already be thrown out, indicating that the idf term will not be the dominating factor.

Finally, the similarity of a query vector Q and a document vector V can be computed via the scalar product of the two vectors:

$$Similarity(Q,V) = \sum_t w(t)_q \times w(t)_v$$

Since the tf-idf information retrieval scheme is simple, it is one of the most popular methods used today. Additionally, the method provides good results. Depending on the data set, very high precision can be achieved in retrieving relevant articles (Salton, 1991).

4.2.2 Global Hill Climbing - Bayesian Induction

The initial classification scheme considered for INFOS to operate upon the extracted features was a Bayesian induction classification method. This method is simple to perform and has been popular in a variety of machine learning settings (Weiss & Kulikowski, 1991). Bayesian induction follows directly from the Bayes rule. The goal is to find the probability of a classification C, given evidence (features) e:

$$P(C|e) = \frac{P(e|C)P(C)}{P(e)}$$

With the assumption of conditional independence, the equation can be transformed into a simple product. Given a set of training data, a table of probabilities can be easily constructed to approximate P(e|C). For example, if we are given an article with two possible classifications (Accepted or Rejected) determined by the single feature of word (Flames, Filtering, or Windsurfing), then a table with the following probabilities might be constructed based upon the training data:

Probability	Class-Accepted	Class-Rejected
P(Class)	0.5	0.5
P(Flames | Class)	0.2	0.8
P(Filtering | Class)	0.9	0.1
P(Windsurfing | Class)	0.6	0.4

Table 2: Probabilities for Bayesian Induction

When given a new object to classify, under conditional independence we simply choose the class that maximizes the following expression derived from the Bayes Rule:

$$weight(t)_i = tf(t)_i \times idf(t)$$

$$Similarity(Q, Class) = \sum_t w(t)_q \times w(t)_{Class}$$

While this method does take into account the frequency of each feature term, it is slightly more difficult for users to understand than a keyword-based exact lookup system. For each feature, the system needs to be able to compute the term frequency per class and the inverse document frequency. To compute the term frequency per class, INFOS requires the number of samples of each class and the number of times the term has appeared in each class. To compute the inverse document frequency, INFOS requires the total number of documents and the number of times the term has appeared in all documents.

The entire computation for both the Bayesian and tf-idf methods involve four variables for users to manipulate (number samples of each class, number of terms appearing in each class, number of total documents, number of terms appearing in all documents). Moreover, these variables are combined into two terms that compete with each other inversely. In psychological experiments, subjects preferred negation/addition strategies over reciprocal/inverse strategies (Collis, 1963). Consequently, if users need to edit their own profiles, linear negation/addition relationships may be preferred over the more complex inverse/reciprocal relationships required in the tf-idf method.

4.3 Global Hill Climbing Algorithm

Based upon the strengths and weaknesses of both Bayesian induction and tf-idf, a simpler scheme has been implemented in INFOS that is inspired by both methods. This method, termed Global Hill Climbing, is a linear discriminant method based on a table of features. This table counts the number of times each feature has been found in each class. Since the table contains only one variable per class, it is simple for users to understand and manipulate. The table is created in a hill climbing fashion; as the user reads messages, she indicates whether or not each message read was accepted or rejected. The outcome is used to increment the table's weights accordingly.

An example is shown in table 3. Here, the feature "genetic" has appeared in five accepted articles, the author feature of "grog@ucdavis" has appeared in three accepted articles and one rejected article, etc. This data indicates an interest in articles posted by grog or containing the word "genetic," and a disinterest in articles containing the word "flames." In addition to using words from the articles as features, collaborative review features are also included in the table. These other users are local users running the same news system who are willing to share their own reviews with others. In table 3, the other user "Renee" has accepted four articles the current reader has accepted, and Renee has rejected one article the current user has accepted. Similarly, Renee has rejected two articles the current user has accepted, and rejected three articles the current user has rejected. This table indicates that the current reader's accepted messages strongly correspond with Renee's accepted messages, while the current user's rejected messages slightly correspond with Renee's rejected messages. The table continues to grow as new articles are read.

Word	Accepted	Rejected
genetic	5	0
algorithm	3	3
flames	2	7
grog@ucdavis	3	1
Renee Accepted	4	1
Renee Rejected	2	3

Table 3: Global Hill Climbing Table of Weights

Given such a table, classification of new messages is performed by extracting the features from the new article and then computing the sum of all the Accepted and Rejected values from matching features in the table. If the Accepted percentage minus the Rejected percentage exceeds A, the message is classified as being of interest. Conversely, if the Rejected percentage less the Accepted Percentage exceeds A, the message is classified as being of no interest. Messages in between are marked unknown. In this project, A was set to 0.15 so that some margin of difference was necessary to classify a message either way. However, this has been left as a user-adjustable setting to allow more aggressive or more conservative classifications to be made. Mathematically, the classification process for a set of feature terms *t* is referenced by:

$$SimilarityPercentage(class)_t = \frac{\sum_t ClassOccurrences_t}{\sum_t TotalOccurrences_t} \quad (1)$$

$$Class_t = \left\{ \begin{array}{c} (SimilarityPerc(Acc)_t - SimilarityPerc(\mathrm{Re}\, j)_t) > A{:}Accepted \\ (SimilarityPerc(\mathrm{Re}\, j)_t - SimilarityPerc(Acc)_t) > A{:}\mathrm{Re}\, jected \\ else{:}Unknown \end{array} \right\} \quad (2)$$

The global hill climbing scheme bears some similarities to a Bayesian approach assuming conditional independence among the features. However, by computing sums, the frequency of occurrence for each feature is considered and a cutoff point is established. The system is closer in similarity to the tf-idf method, but it does not

explicitly reference the inverse document frequency. However, this term contributes only a small amount compared to the tf term in tf-idf. Moreover, the purpose of the idf term is to remove non-predictive words. Most of these words can be filtered out early on through the use of the stop list. Finally, even if words that have a large index-document frequency enter into the global hill climbing table, those words that are non-predictive will still have little impact since the accept/reject probabilities will be approximately equal and cancel each other out. For example, the word "the" is not biased towards rejected or accepted articles, and although it will have a high frequency of occurrence, it will not be a factor in classification since both the rejected and accepted categories will contain approximately equal occurrences of the word. As a result, little is lost and the user profile is simpler, making user manipulation an easy task.

An example of editing the user profile is depicted in figure 12. Editing is as simple as incrementing or decrementing the accepted or rejected values.

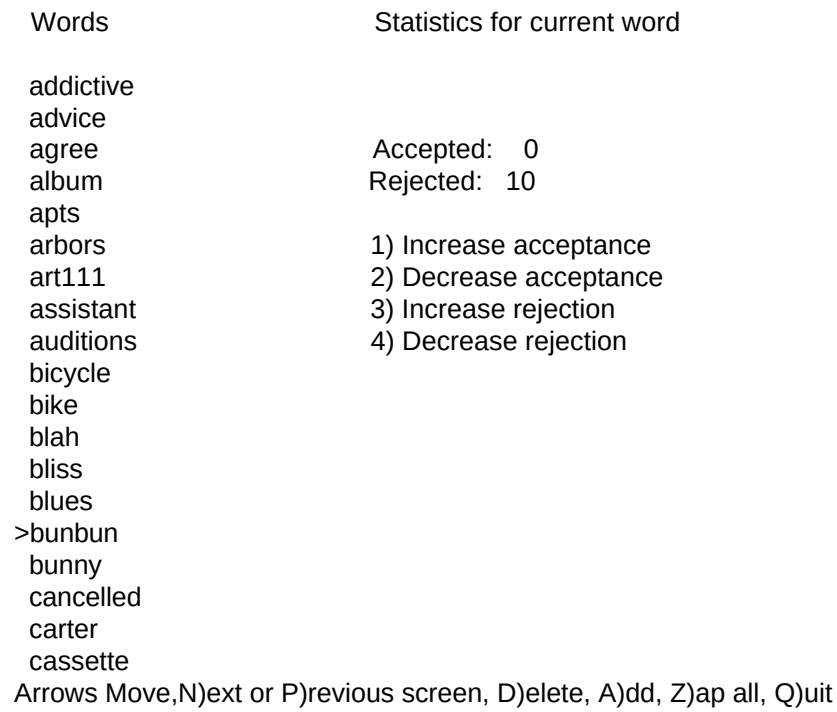


Figure 12 : Sample of INFOS edit screen for global hill climbing method

The currently selected word, “bunbun”, has a high rejection value indicating that this term often appears in articles the user is not interested in. In the event that a user wishes to search for specific features, the user simply sets up increments the weights for features of interest and turns learning off so that no new data is added to the table. However, filtering will continue using the user-specified data.

4.4 Assigning Weights

As the algorithm stands, all features are treated equally. Authors, text from the body, text from the subject, and collaborative data are all counted and combined in the same way. While this allows each feature to account for as large or small a contribution as desired, this method is biased to favor those features that occur most often. For example, the word “computer” is much more like likely to occur in the body of articles in a computer newsgroup, than the author of a particular group. The computer term may appear hundreds or thousands of times, while an individual author will probably only appear a handful of times. As a result, the contribution from author’s terms will be negligible when compared against other more frequently occurring features.

One solution to this problem is to separate the global hill climbing table into a set of individual tables - one table for each type of feature. Percentages of acceptance and rejection can be computed from the features among each table, and then these percentages combined to compute the final classification:

$$SimilarityCombn(Class)_t = \begin{array}{c} K_1 \times SimilarityPerc(Class)_{author} + K_2 \times SimilarityPerc(Class)_{sub} + \\ K_3 \times SimilarityPerc(Class)_{text} + K_4 \times SimilarityPerc(Class)_{collaborative} \end{array}$$

$$Class_t = \left\{ \begin{array}{c} \left(SimilarityCombn(Acc)_t - SimilarityCombn(\mathrm{Re}\, j)_t\right) > A{:}Accepted \\ \left(SimilarityCombn(\mathrm{Re}\, j)_t - SimilarityCombn(Acc)_t\right) > A{:}\mathrm{Re}\, jected \\ else{:}Unknown \end{array} \right\} \qquad (3)$$

However, how should these percentages be combined? What values should be assigned to constants K_1 through K_4? Some systems (Jennings & Higuchi, 1992) give higher weight to the subject features on the assumption that these are most predictive. To investigate which terms are actually most predictive, experiments were performed to evaluate the impact of each feature individually: classification using author features alone, subject features alone, text body features alone, and collaborative features alone. All features were then combined based upon how much impact they showed individually; i.e., the most predictive feature was given the highest weight, and the least predictive features given the lowest weights.

4.5 Global Hill Climbing - Experimental Results: Varying Filtering Features

To test how the different features contribute towards making relevant classifications, the global hill climbing scheme was tested using each feature set alone. The data used for the tests were the same messages extracted from the ucd.life newsgroup read by the 14 subjects. Each user read 100 sequentially posted messages and marked each as accepted, rejected, or unknown. From these 100 messages, 50 messages were randomly selected for training, and the system predicted the users' choices for the rest of the messages using equation 2 only among one set of features. These predictions were one of three classes: Suggested, Not Suggested, or Unknown. These predictions were then compared to the actual classifications provided by the subjects. Samples of the data articles and the domain are given in chapter 2. The evaluation metric used in this experiment is classification accuracy. In INFOS, accuracy is defined as the percentage of predicted articles that were classified correctly.

The experimental results are shown in table 4. This table contains the percentage correct, unknown, and incorrect classifications averaged over the 14 test subjects. If a

prediction matched the response provided by the user, then that article was marked as being correct. However, if the prediction was unknown, then that article was marked as being unknown. Finally, if the prediction was not unknown but incorrect, then the article was marked as being incorrect. This classification scheme rewards INFOS for being correct, penalizes it for being incorrect, but neither awards nor penalizes when the system produces an unknown class. The results show that the subject features results in the highest percentage correct (52%) with the lowest error (12%), probably since subject words are accurate predictors of entire threads that may be of interest. The textbody features actually give the largest percentage correct (54%), but also give the largest error (19%). Collaborative filtering gave the next best results (46%), and author alone was the worst predictor (38%), although not far behind the others. Note that all schemes perform better than chance or by always predicting the most likely class. Overall, 36% of the articles were rejected, 35% accepted, and 29% unknown. A trivial classifier guessing by chance would have 33% accuracy, and a trivial classifier always predicting the most likely class would have 36% accuracy, but 64% error. Consequently, all methods are performing non-trivial classifications.

Features Used For Classification	**Percentage Classified Correctly**	**Percentage Classified Unknown**	**Percentage Classified Incorrectly**
Author Alone	38.4	46.7	14.9
Subject Alone	52.1	35.5	11.8
Textbody Alone	53.6	27.2	19.2
Collaborative Alone	46.2	41.2	12.6

Table 4: Classification accuracy for individual sets of features.
Results are averaged over 14 subjects, showing percentage classified correctly, incorrectly, and unknown for 100 consecutively posted articles, 50 articles read.

The results from this experiment indicate that the subject features should have the highest weighting, followed by textbody and collaborative data. Author features should have the lowest weighting. A value of 0.35 was assigned to K_2, the subject's weight,

0.25 to K_3 and K_4, the collaborative and textbody weights, and 0.15 to K_1, the author's weight. These weight settings correspond to the performance of each feature set. The textbody weight was lowered to 0.25 since this set did have a higher error rate than all other methods.

4.6 Global Hill Climbing - Experimental Results: Combined Features

Using the weights determined from the previous study, the classification process was run again using the new weights and equation 3. The results are shown below in figure 13:

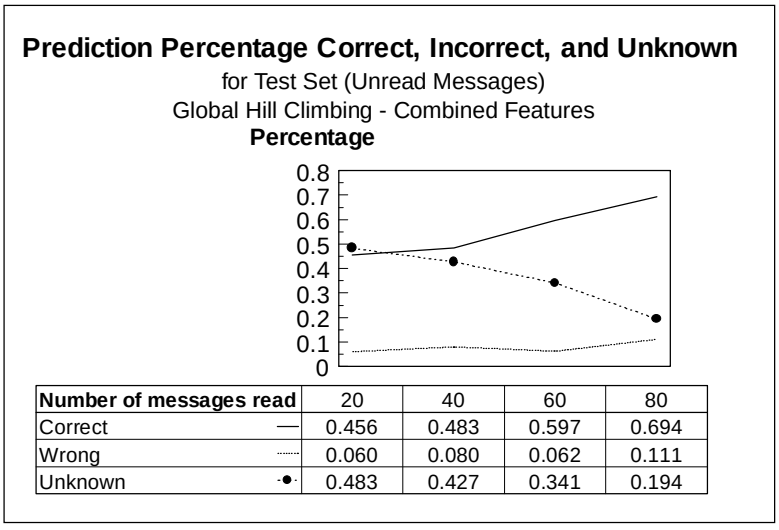
Percentage Correct Classifications	51.5%
Percentage Incorrect Classifications	7.3%
Percentage Unknown Classifications	40.9%
Within Error, Percent of False Positives:	50%
Within Error, Percent of False Negatives:	50%

Figure 13 : Classification results for combined global hill climbing scheme

The percentage of correct classifications, 51.5%, is slightly lower than using the subject scheme alone, but the error is significantly smaller. At 7.3%, the error is lower than using any of the feature sets alone. Combining the set of features results in overall improved performance. The types of errors that are made are evenly split among false positives (an article predicted to be of interest, but really is not of interest) and false negatives (an article predicted to be rejected, but really is of interest). Typically, false negatives are the worst of the two types of errors; false positives may result in users becoming annoyed at reading material they do not want to read, but false negatives may result in users missing a message of interest entirely.

4.7 Global Hill Climbing - Varying Number of Messages Read

The previous experiments all trained upon 50 randomly selected messages. An experiment was also conducted that examined accuracy when 20, 40, 60, and 80 randomly selected messages were used in the training set. The results are shown in figure 14. Not surprisingly, as more messages are read, the percentage of correct classifications increases from 45% to 70%, and the percentage of unknown classifications decreases from 48% to 19%. The error stays relatively constant. The main result of this experiment is that the performance does not vary significantly from 20 messages read to 40 messages read; not until half or more messages are read does performance increase noticeably. Since users will want to achieve maximum performance while reading as few messages as possible, then as long as messages are randomly selected, reading as few as 20% of the unread articles still gives adequate performance. Not many messages need to be read in order to train the system since the tokens just processed give a good indication of what other topics are currently being covered.



Number of messages read	20	40	60	80
Correct	0.456	0.483	0.597	0.694
Wrong	0.060	0.080	0.062	0.111
Unknown	0.483	0.427	0.341	0.194

Figure 14 : Performance versus number of messages in training set

4.8 Global Hill Climbing Performance - Completely New Articles Threads

Another parameter that has not yet been examined is the performance of the system on completely new messages. The tests so far have been performed on a sequence of messages that have been posted sequentially. As a result, the topics among these messages are closely correlated; many of these articles occupy the same threads. This comprises normal operation if the user keeps up by reading new articles every few days. However, if the user fails to read messages for a few days or weeks, then completely new article threads and conversations will be posted.

To examine the impact upon performance under this scenario, subjects read 100 consecutively posted messages to build up a user profile. The system was trained upon the features from these messages and then tested on 50 unread messages that were posted two months later. These future messages were comprised of completely different threads, although some of the material was also discussed in the original set of messages.

The results from this experiment are shown in table 5. While the subject features provided the best results in the previous experiment, the subject features provided the worst correct classification rates in this experiment. This is due to the new threads present in the test set; subject tokens are excellent predictors of threads, but with entirely new threads this feature is not nearly as predictive. However, words from the body of articles are not predictors of threads. Consequently, the textbody features give a much higher percentage correct, but also result in a very high error. The best features turn out to be the *indirect* features of author and collaboration. Both of these features do not use semantic data. With an entirely new set of messages, the previously processed semantic data is not as useful; however, features that are not semantic will still be valid. As a result, the author and collaborative features produce the lowest error and acceptable correct classification rates.

Features Used For Classification	Percentage Classified Correctly	Percentage Classified Unknown	Percentage Classified Incorrectly
Author Alone	20.4	76.0	3.3
Subject Alone	13.1	86.3	0.6
Textbody Alone	45.4	28.9	25.5
Collaborative Alone	36.3	53.4	9.1
Combined Features	30.9	59.8	7.7

Table 5: Classification accuracy for individual sets of features.
Results are averaged over 14 subjects, showing percentage classified correctly, incorrectly, and unknown for 50 posted articles from new threads, 100 articles read.

This data suggests that when reading completely new threads unrelated to the material last read, the non-semantic features of author and collaborative features should be given the highest weight in order to minimize error. Based upon these results, the collaborative and author terms should be high, while the textbody and subject lower to minimize error. The experiment was re-run using values of 0.15 for K_2, the subject's weight, 0.30 for K_1, the author's weight, 0.35 to K_4, the collaborative weight, and 0.20 to K_3, the textbody's weight. The results are shown in table 5 in the Combined Features row. As expected, the correct classification (30.9%) is lower than when filtering is performed using training data from the same set of articles as the test data, but the error remains relatively low at 7.7%. These results indicate that filtering is still feasible for completely new sets of data, but first the weights should be adjusted to favor non-semantic features, and also accuracy will be lower.

4.9 Global Hill Climbing - Variable Weight Scheme

While the weighting system allows rarely occurring feature sets to contribute to the classification rather than be dwarfed by more frequently occurring feature sets, one limitation of this scheme is that each set of features is also limited in the overall contribution it can make. For example, with a weight of 0.15 assigned to the author

feature, no matter how important a particular author may be, any set of values assigned to the author's weights cannot contribute more than 15% to the total. This may result in messages being missed if the other combination of features is not enough to classify the article. If the user wants to read all articles posted by a particular author, no matter how uninteresting the other features may be, then the weight limitation may be a problem.

A simple solution is to lower the threshold value for accepted articles. If this value is lower than 15%, then the author contribution is more likely to have an impact on classification. However, other features may still compete with the author feature, resulting in an undesired classification. To examine the effect of weighting limitations, an alternate system of combining feature values was explored. In this method, each feature *x* (author, subject, textbody, or collaborative set) was scaled so that if all occurrences were encountered, the total is one:

$$Unit_x = \frac{1}{TotalAccept_x + Total\,\mathrm{Re}\,ject_x}$$

At this point, computing a classification via summed scaled units of features instead of the original feature counters results in each feature being treated equally. In terms of frequency of occurrence, all features are scaled appropriately. Features that appear often (such as textbody tokens) will have very small unit values, while less frequent features (such as authors) will have larger unit values. Consequently, a single feature is no longer limited to a predetermined contribution to the overall classification, but may have a large impact if its values are adjusted large enough.

Based upon the results obtained in section 4.5, the unit values were also multiplied by scaling factors so that more predictive features are given a higher weight. Subject units were multiplied by a scaling factor of 3, collaborative units by 2, textbody units by 2, and authors by 1. Note that these scaling factors bias the filter to consider certain features more important than others, but does not limit the contribution of other features to a predetermined percentage. For example, the author weights can still contribute more to

the final classification than any other features, but the actual author values must be three times larger than the corresponding subject weight.

Results using the variable weighting scheme upon both consecutively posted messages and completely new message threads are shown in table 6. The scheme had a 7% higher error rate than the fixed-weighting scheme on the consecutive messages. A higher error rate is to be expected since poor predictors are now allowed to have a larger impact on classification. Results for the dataset with separate threads is slightly better than the weighted scheme, although not significantly. This data indicates that overall, the variable scheme does not perform much better than the fixed-weighting scheme but may still be useful for directed user search.

Features Used For Classification	**Percentage Classified Correctly**	**Percentage Classified Unknown**	**Percentage Classified Incorrectly**
Consecutive Messages	55.2	29.5	15.0
New Message Threads	37.1	54.6	8.3

Table 6: Classification accuracy for individual sets of features.
Results are averaged over 14 subjects, showing percentage classified correctly, incorrectly, and unknown for both consecutively posted articles and completely new articles.

4.10 Chapter Summary

This chapter has described the global hill climbing algorithm implemented in INFOS. The highlights of this scheme include:

- News headers, encoded files, and quoted material are currently ignored by INFOS. Although the determination of quoted material is difficult, simple algorithms implemented in INFOS incorrectly detected quoted material with only 2% error.

- The global hill climbing method is based upon a simplified version of tf-idf and Bayesian induction. The simplifications make the model easy for users to understand and modify, but the system still retaining flexibility in classification.

- Classification through keywords extracted from the subject were the most predictive with an accuracy rate of 52% and an error rate of 12%. When combined with author, textbody, and collaborative features, the accuracy rate remained constant at 52% but the error rate dropped to 7%.

- As more messages are read and the user model becomes more accurate, the accuracy of the classifier increases while the unknown classifications decrease.

- When reading new messages in a new context, non-semantic features such as the author and collaborative reviews provide the highest accuracy rates for classification.

5. Case-Based Reasoning Method

The global hill climbing method's main strength lies in its simplicity, user modifiability, and predictive abilities for features that have been previously encountered. In this case, the system will build a compact representation of user interests. However, the global method does have some major weaknesses. First, the global method is unable to discern fine differences in features because it linearly combines all input features through the conditional independence assumption; e.g., if we are not interested in messages with the features "dynamic" and "algorithms" but we are interested in messages with the features "genetic" and "algorithms", then the global method will be using the same accepted and rejected values for the word "algorithms" and may be unable to classify correctly one or both classes of messages. Second, the global method has no semantic content about the meaning of words. The system will make separate table entries for the words "bicycle" and "bike" when these words are really referring to the same thing.

Some solutions that address the first problem include neural network and genetic algorithm schemes. Both of these were examined, and the genetic algorithm method is described in chapter 6. Preliminary experiments were also performed using a radial-basis function neural network to classify messages. In this approach, the extracted features were presented to the input units and the output units determined whether or not the article was accepted or rejected. Although more powerful than the global hill climbing approach since it is able to combine non-linearly any of the input features in making a classification, this approach did not give better results than the global hill climbing method and has the further drawback that a user cannot easily modify the weights of a network to control classifications directly. Moreover, training time for the network was long, further limiting the usefulness of such a system. For these reasons, a neural-network based

scheme was abandoned. However, training of a neural network upon large data sets may ultimately provide better results and requires further investigation.

The method used instead in INFOS is a case-based reasoning system. A case-based scheme is capable of addressing the weaknesses of the global hill climbing method while also providing additional functionality. By retrieving individual cases and using the classification of those cases as the new classification, the system is capable of avoiding the limitations of linearity. Furthermore, by designing a case-based reasoning system with semantic knowledge, INFOS is capable of comparing concepts rather than individual words. Additionally, a CBR system also provides an excellent opportunity to support information retrieval in addition to information filtering. Finally, other systems based upon this type of case-based technology have been successful (Alvarado et. al., 1993; Alvarado & Mock, 1995).

The case-based reasoning component of the system has three major sub components: the method for extraction of indices relating to the document, the method in which articles are stored using these indices, and the method in which articles are retrieved and classifications made. All of these factors are discussed and then experimental results given for the case-based scheme alone when used alone and when combined with the global scheme.

5.1 Index Extraction

To index cases, noun-phrases representing the key concepts of individual sentences are extracted and used with other features such as the author, collaborative data, etc. to perform the actual indexing. As with the global hill climbing method, the article is first passed through a stop list to remove common, non-predictive words. Headers, binary encoded files, and quoted material is also removed. Afterwards, the article is converted to all lowercase for extraction of concepts.

This work uses both controlled and uncontrolled index extraction. In the controlled approach, a predefined list of terms or knowledge structures is used to guide the indexing process. This approach is powerful and can detect concepts with high accuracy; however, one most have a fully defined knowledge base and predict all the structures that may occur. Currently, this is not possible for new domains. The uncontrolled approach relies on general purpose methods rather than pre-existing domain knowledge to create indices. As a result, indices may not be specific or well-defined as the controlled approach, but the benefit is generality across all domains. This project uses a combination of both approaches in an attempt to get the benefits both schemes offer. The controlled approach in INFOS is composed of a knowledge-based method derived from WordNet, while the uncontrolled approach is composed of a keyword based method.

5.1.1 WordNet and Index Extraction

Index extraction is performed in a manner similar to the procedure described by Paice (Paice, 1989) to automatically extract text phrases for inclusion as back-of-book indexes. However, the use of WordNet is utilized to map words to concepts, and these concepts are used as indices rather than the actual words (Miller, 1995). In the event that a word is missing from the WordNet lexicon, then that word is used in an inverted index to index the source document directly. To narrow the amount of data required for processing articles, INFOS only focuses upon the verbs and nouns indexed in WordNet. This approach is supported by the indexing community since indices for books are almost always based upon nouns and verbs instead of adjectives, adverbs or other parts of speech (Evans et. al., 1991).

WordNet is a project at Princeton University to create a knowledge-base of English words which include part of speech identification, word usage, synonyms, frequency usage, attributes, meronyms, and hyponyms of words and was created with the

intent that it can be useful for natural language projects. Nouns are defined in terms of a hierarchical semantic organization; e.g., the word “oak” is defined as a oak-->tree-->plant-->organism, where arrows indicate ISA relationships. While WordNet contains additional information about words, such as PART-OF (meronym) relationships, the mapping of a noun or verb into the ISA hierarchy is the main use of WordNet in INFOS. Since the current version of WordNet (v1.5) contains approximately 107,000 noun senses and approximately 27,000 verb senses - approximately the size of a paperback dictionary - WordNet is capable of recognizing terms from a broad variety of topics. This should eliminate the need to create specialized knowledge bases of limited domains and allow processing in a variety of different domains. Domain specific keywords, indexed directly, can server to further differentiate different domain articles.

An example of the WordNet ISA hypernym hierarchy for the word “ocean” is shown in figure 15. When a word is found in the WordNet lexicon, all definitions or *senses* of that word are provided. In the case of ocean, there are two noun definitions; one for the body of water, and the other indicating a large quantity. These definitions are organized hierarchically, from the most specific up to more abstract concepts.

```
Sense 1
main, ocean, sea, briny
      => body of water, water
         => object, inanimate object, physical object
            => entity

Sense 2
ocean, sea
      => large indefinite quantity
         => indefinite quantity
            => measure, quantity, amount, quantum
               => abstraction
```

Figure 15 : Example WordNet hypernym hierarchies for the word “ocean.” This word has two sense definitions, organized from the specific to the general.

The heart of the approach to automatic index extraction lies in the problem of finding appropriate noun phrases or verb phrases. Previous work (Evans et. al., 1991;

Paice, 1989) has showed that noun phrases can be effective indices into articles. By limiting the indexing to noun phrases, this simplifies the natural language parsing process. While this process certainly loses a large amount of linguistic information required for detailed natural language processing, in this work an assumption has been made that most of this information is not necessary for information filtering.

5.1.2 Topic Neighborhoods to Identify Key Phrases

One of the keys to identifying key noun phrases is identifying the potential phrase heads (Salton et. al, 1971). A variety of ad hoc rules have been useful in the past to locate key noun phrases, including location (first two and last sentences of a paragraph tend to be important), headings, captions, cue expressions such as "important," or transition words such as "however" (Edmundsun, 1969). Rather than use these potentially unsafe strategies, the approach taken in INFOS relies more heavily on statistics.

A useful observation regarding text is that topics do not change abruptly. In particular, with short news articles, many entire articles have one central concept. However, articles may cover different topics, but typically the topic changes bit by bit as new material replaces old material (Paice, 1989). This supports a theory of "topic neighborhood" - sentences of the same topic tend to be located near each other within some sentence radius neighborhood. The topic neighborhood theory may be exploited to find good index words. Words that make good indices and are representative of an article are terms that are not commonly used in English, but are common in the text; those with a high frequency are likely to be relevant of the topic at hand. However, concepts are not frequently referred to explicitly by the same name, but often by their logical components. As an example, consider the following:

> "There is nothing wrong with this [knowledge representation] model as far as it goes. But at the same time anyone familiar with AI must realize that the study of

> knowledge representation - at least as it applies to the 'commonsense' knowledge required for reading typical texts such as newspaper - is not going anywhere fast. This sub field of AI has become notorious for the production of countless non-monotonic logics and almost as many logics of knowledge and belief, and none of the work shows any obvious application to actual knowledge-representation problems. Indeed, the only person who has had the courage actually to try to create large knowledge bases full of commonsense knowledge, Doug Lenat, is believed by everyone save himself to be failing in his attempt... it is therefore time to switch paradigms. (Charniak, 1993, pp xvii)."

In this paragraph, "knowledge representation" is a relevant term. It is repeated once, but there are cases where "commonsense knowledge" or "knowledge bases" are used, and reference the same topic. Consequently, Paice's hypothesis to find relevant topic words are to identify *word stems* that frequently occur close together in a sentence. These word stems form *topic words* or significant phrases. This approach can be augmented by finding *topic synonyms* that frequently occur close together in a sentence, rather than exact word matches. Word synonyms can be easily found, along with identifying parts of speech, through WordNet.

To accomplish these tasks, verbs and nouns from each sentence are first identified through WordNet and their hierarchical definition referenced. This step results in a linked list of nodes, where each node contains the hypernym sense definition for the nouns and verbs in that sentence. Sentence and word definition order are maintained to allow for other forms of processing, such as scripts or syntactic rule processing described in chapter 7. Since each word is expanded into all possible sense definitions of that word, this pool of sense definitions may not accurately reflect the actual topic. For example, in the sentence "the ocean is cold.", both definitions of ocean from figure 15 will be expanded into the node list. However, only the body of water definition is relevant; the large-quantity definition does not apply.

A simple method to disambiguate the correct word sense is to select the sense that occurs most frequently (Miller et. al., 1994). However, this scheme obviously suffers when the most frequent sense is not referenced. Furthermore, the possibilities of increased

performance are limited since most senses will never be considered. The approach implemented in INFOS expands a word into all senses and then attempts to disambiguate and eliminate senses that appear irrelevant.

5.1.3 WordNet Based Index Extraction Algorithm

To refine the sense definitions and select relevant ones, neighborhoods of sentences are examined and the intersection of sense definitions that match within a specified neighborhood are selected. This process restricts the selected definitions only to those that are reoccurring topic stems and are then more likely to be relevant to the document. To compute these stems, a modified version of Paice's algorithm was implemented in INFOS. This algorithm computes topic senses for each sentence S_0 of an article is shown below in figure 16. The notation uses S_0 to denote the current sentence, S_{-1} to denote the previous sentence, T_0 to denote the current sentence's topic concepts, T_1 to denote the next sentence's topic concepts, and so on. Only the first 20 sentences of articles were processed to speed execution in the event of extremely long postings. The assumption was also made that long messages will contain relevant material at the beginning of the article.

```
Process text article through stop-list, strip quoted material
for each sentence S0 up to 20 sentences in the text do
begin  { Find senses to use as indices }
   for each successive word W in S0 do
   if W is a verb, noun or part of a noun or verb phrase
   begin
          find set of ISA hierarchy meanings, {M0}, for word W via WordNet
          if any topic from {M0} is a topic of the previous sentence's topics {T-1},
                i.e. any x ∈ {M0} and x ∈ {T-1}
          then add x to {T0}  (Add matching topics from
                                last sentence to current)
          else if no x∈ {M0} was not a topic of previous sentence or
                incidence {M0} for next six sentences > SwitchOn
          then assign {M0} as topic for current sentence {T0}  (Add new topic)
   end
   else  add word to inverted index    (Word not found in WordNet)
   for each x ∈ {T-1} do   { Assign sense topics to sentences }
   begin
          if x ∉ {T0} and incidence x for next ten sentences  > SwitchOff
          then add x as topic of {T0}; i.e. add x to {T0}
                (Add topic based upon neighborhood of nearby sentences
                 if not mentioned explicitly in current sentence)
   end
end
```

Figure 16: Modified Paice's algorithm to compute topic indices

In this algorithm, the incidence indicates how often a topic T occurs in a sentence S. This is computed by:

$$\text{Incidence}(T,S_i)=\sum_{k=1}^{n}\frac{1}{2^{k-1}}$$, where n is the number of occurrences of T in S_i.

Consequently, if T occurs once in S, I=1, if T occurs twice in S, I=1.5, and if T occurs three times, I=1.75. This equation simply scales the incidence to a value between 1 and 2, where 1 indicates one occurrence and 2 indicates many occurrences. The SwitchOn and SwitchOff parameters were initially set to 1.5 so that at least two occurrences were necessary within the given neighborhood in order for concepts to be activated.

This algorithm attempts to assign topics to each sentence in the text, taking into account the frequency of topic sense throughout the entire document, while also maintaining a "neighborhood radius" if the same topic applies to sentences before or after the current sentence. Consequently, for each topic sense, a corresponding range of relevant sentences is computed. Although not implemented in INFOS, this information can be stored to indicate which sequences of sentences corresponds to a particular topic. This information could be useful when retrieving messages to show what sentences are relevant to a new query.

In the event that none or very few (less than four) topic senses are identified through this procedure, and the total number of sentences is small (less than five), then the article may be too short for the topic sense algorithm to perform well. Since some features are needed for indexing, INFOS simply selects up to an additional 15 sense definitions classified as uncommon to use as indices. The frequency of occurrence of a sense, or the commonality, is an attribute of each sense that is accessible through WordNet polysemy feature.

Once candidate nouns and verbs have been identified and assigned to sentences, this information is used to index the document. In addition to the sense definition itself as an index, other relevancy statistics are also associated to each term, including frequency and rarity (Evans et. al, 1991). *Frequency* is merely the number of times the term appears in the document / number of times the term has appeared in the domain. This measure operates upon the assumption that domain words appearing often are indicative of the document. *Rarity* is a measure of the expected frequency of a word in general English. In WordNet, this is obtainable through a terms polysemy count. A common word such as "system" has a high value of 15, while a rare English word such as "cilia" has a low value of 2. In INFOS, the rarity R is defined as:

$$R_{sense} = 1 - \left(\frac{Polysemy_{sense}}{Max\_Polysemy} \right)$$

Based on this definition, extremely rare words will have a value of 1, while common words will have a value closer to 0. Note that extremely common words will not be considered, as they will be filtered out before processing.

Once both frequency and rarity have been determined, the two are multiplied together to give a general relevancy statistic for a sense term:

Relevancy $R = Rarity \times Frequency$ for each term. (4)

The relevancy value is stored with each term and is used in memory retrieval to determine how closely an old article matches a new document.

5.2 Indexing of Cases

Once the appropriate noun and verb phrase senses have been extracted from a textual case, the article is saved and the senses used to index the case. In cases where concepts cannot be extracted (e.g., with novel words or domain-specific slang words and expressions) these terms are used directly to index the case through an inverted index. In this way, cases are indexed via both conceptual (controlled) and keyword (uncontrolled) vocabularies, allowing conceptual retrieval when possible, and also keyword retrieval under unforeseen situations so that performance is still possible (Callan & Croft, 1993). Figure 17 depicts a sample inverted index. Based upon a word or token, references are made to all case articles that contain the token. In INFOS, each case pointer also contains the number of hits and the number of times that case has been accepted and rejected for the given token to compute relevancy. Furthermore, the case itself has a classification of accepted or rejected to compute a classification for similar cases. This example also shows one of the weaknesses of the inverted index approach; misspelled words or nonsense

words are included in the indexing process. In terms of efficiency, the index tokens can be accessed through a hash table, resulting in constant time to find relevant articles. However, for purposes of simplicity, the index was implemented as a sorted array in the INFOS prototype. Through binary search, lookup may be performed in logarithmic time.

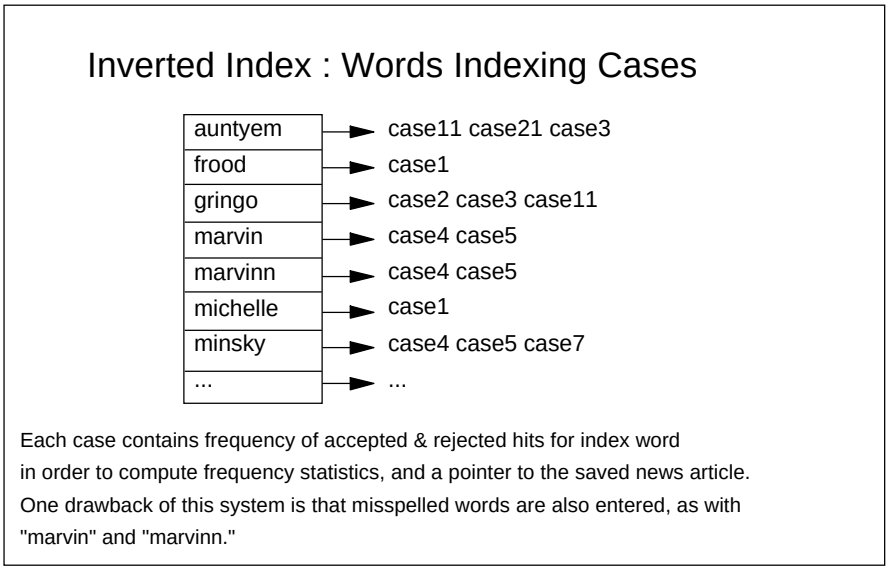


Figure 17: Sample Inverted Index used in INFOS

The method in which articles are indexed using the sense definitions is inspired from Kolodner's case-based memory indexing scheme implemented in the CYRUS system (Kolodner, 1983; Kolodner 1988) and refined in the FANSYS system (Alvarado et. al., 1993). Kolodner's method creates a case-based abstraction hierarchy of Memory Organization Packets (*MOPs*), where a MOP represents a concept. In this hierarchical memory, MOPs are object-oriented. A MOP indexed below another MOP is a specialization of that parent MOP. Furthermore, each MOP contains a set of norms that contain generalized information relating to all MOPs below it. Finally, each MOP is indexed based upon differences. Ultimately, the cases, or *instances*, are found at the bottom of the hierarchy. In this manner, the hierarchy moves from the general (at the root) to the specific (case articles) at the leaves. The advantages of this approach include :

(1) generalization across cases, (2) encoding specificity by grouping similar cases together, (3) fast case retrieval by limiting memory traversal to relevant indices, and (4) elaboration strategies that dictate where in memory to search if the initial search query fails.

The WordNet hypernym hierarchies for the words “vehicle”, “bicycle”, and “car” is given in figure 18. An example memory hierarchy indexed with these concepts is shown in figure 22. In figure 22, one article contains the word “vehicle,” another article contains the word “bicycle,” and the last article contains the word “car.”

Sample WordNet Hierarchies

The WordNet hypernym hierarchy for vehicle is represented as:

=> vehicle
=> conveyance, carrier, transport
=> instrumentality
=> artifact, article, artifact
=> object, inanimate object, physical object
=> entity

Similarly, the concept for bicycle is represented as:

bicycle, bike, wheel
=> cycle
=> wheeled vehicle
=> vehicle
=> conveyance, carrier, transport
=> instrumentality
=> artifact, article, artifact
=> object, inanimate object, physical object
=> entity

Finally, the concept for car is represented as represented as:

car, auto, automobile, machine, motorcar, motor car
=> motor vehicle, automotive vehicle
=> vehicle
=> conveyance, carrier, transport
=> instrumentality
=> artifact, article, artifact
=> object, inanimate object, physical object
=> entity

Figure 18 : WordNet hierarchies for “vehicle”, “car”, and “bicycle.”

All of the concepts in figure 18 are similar in that they are all vehicles; however, the bicycle and the auto differ since one is a wheeled vehicle and the other is a motor vehicle. To index these concepts in the memory hierarchy, a global hierarchy is

constructed that indexes case articles when appropriate, or sub-MOPs that lead to the case articles. In figure 22, the root MOPs are not shown, but the sub-hierarchy starting at the Conveyance concept is displayed. This MOP represents the concept regarding items of transport and conveyance. All sub-MOPs inherit the norms of their ancestors, hence all MOPs located below this must also refer to transportation vehicles. One specific index currently exists from the Conveyance MOP, and it points to the Vehicle MOP. In turn, the Vehicle MOP points to sub-MOPs, one regarding cycles and another regarding automobiles. In addition to pointing to sub-MOPs, the Vehicle MOP also has an index to a specific case referencing vehicles. In a similar fashion, indices from the wheeled vehicle and the auto MOPs are further specialized until they too point to actual cases referencing those concepts.

Kolodner's algorithm to create this memory hierarchy as instances are incrementally encountered is shown in figure 19. The process is straightforward - memory is traversed for each index in the hierarchy until an appropriate location is found to add the case.

Memory Creation Process

When adding a new case C to memory, given new sense indices I of the article:
A) Set the current mop, Cur, to be the root mop. Compare most general terms of sense of I to indices of Cur.

- If no matches result, create new sub-MOPs based upon differences from Cur and I that index the new case. Link these indices starting from Cur and set norms to terms from I.
- Else, for all matching indices from Cur, repeat at step A with Cur set to the matching sub-Mop using the next specific terms of I.
- If last line of terms in I, set Cur to index the case.

Figure 19: Memory Creation Algorithm

Once indices have been selected, memory is traversed in a depth-first, recursive manner along matching indices. During this process, if no indices match, then new MOPS are created that correspond to the new index, ultimately indexing the new case. The new case is always indexed from the current MOP using differences between the

MOP's norms and the new case. Since this is done for all MOPs along the depth-first search path, there may be many MOPs and paths that are linked to the new case.

An example of adding the three indices corresponding to three separate cases is shown in figures 20 through 22. One case contains an index corresponding to vehicle, another case corresponding to bicycle, and the final case contains an index corresponding to car.

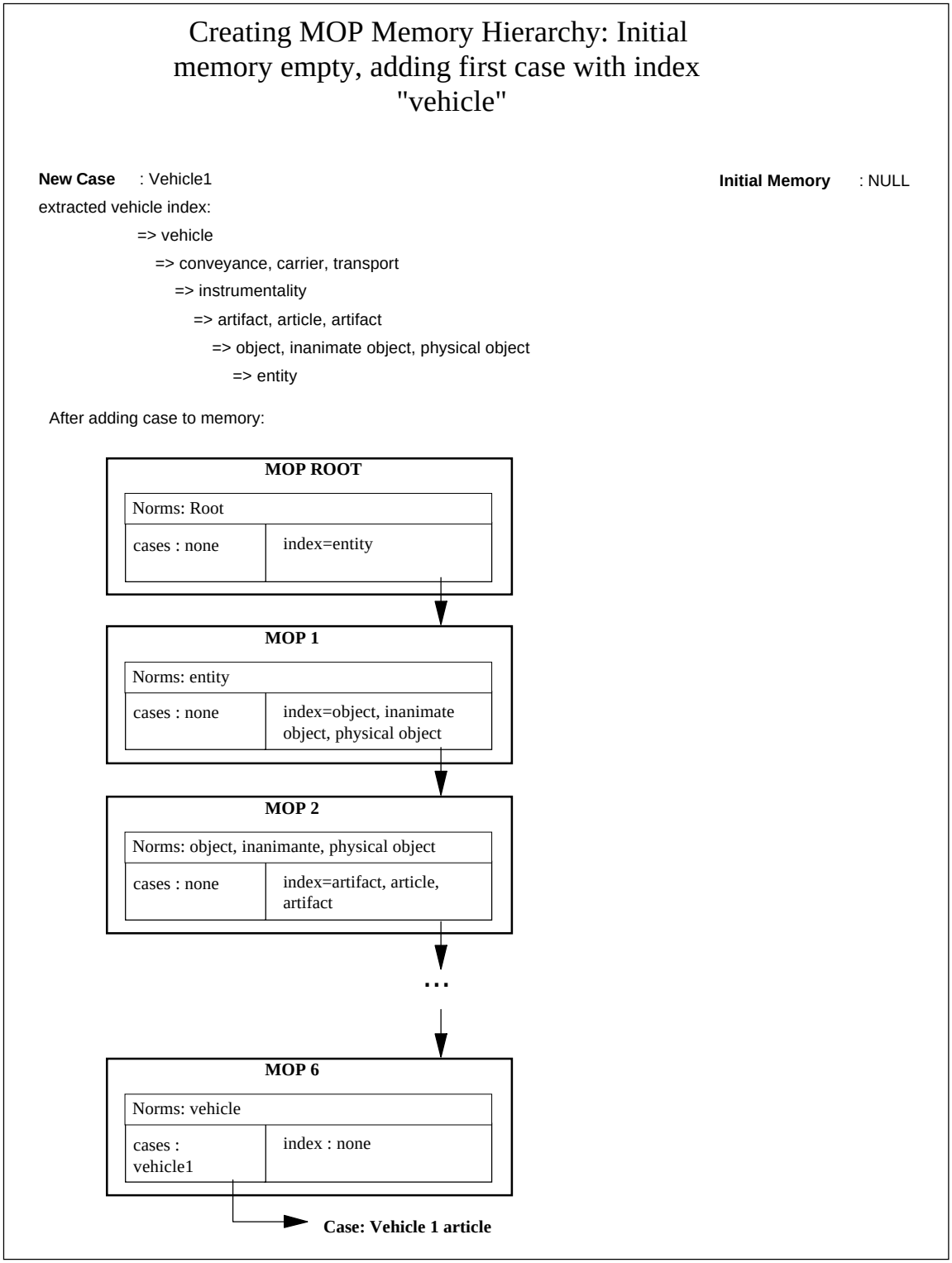


Figure 20: Adding first case to memory regarding vehicle. Case indexed at bottom of hierarchy.

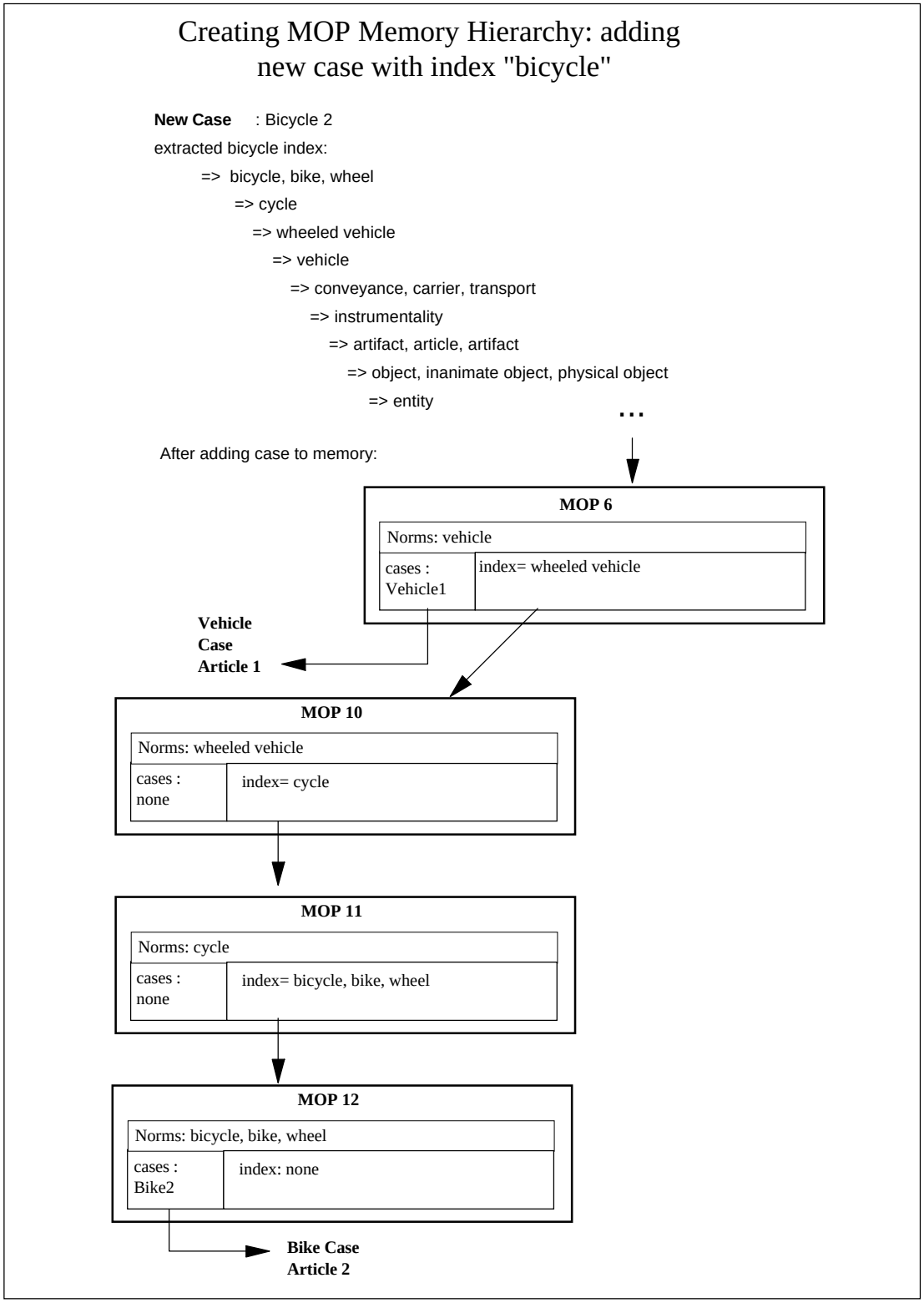


Figure 21: Adding second case to memory regarding bicycle. The bicycle case is added to the portion of the hierarchy matching the vehicle, then new specializations added to the bottom of the hierarchy.

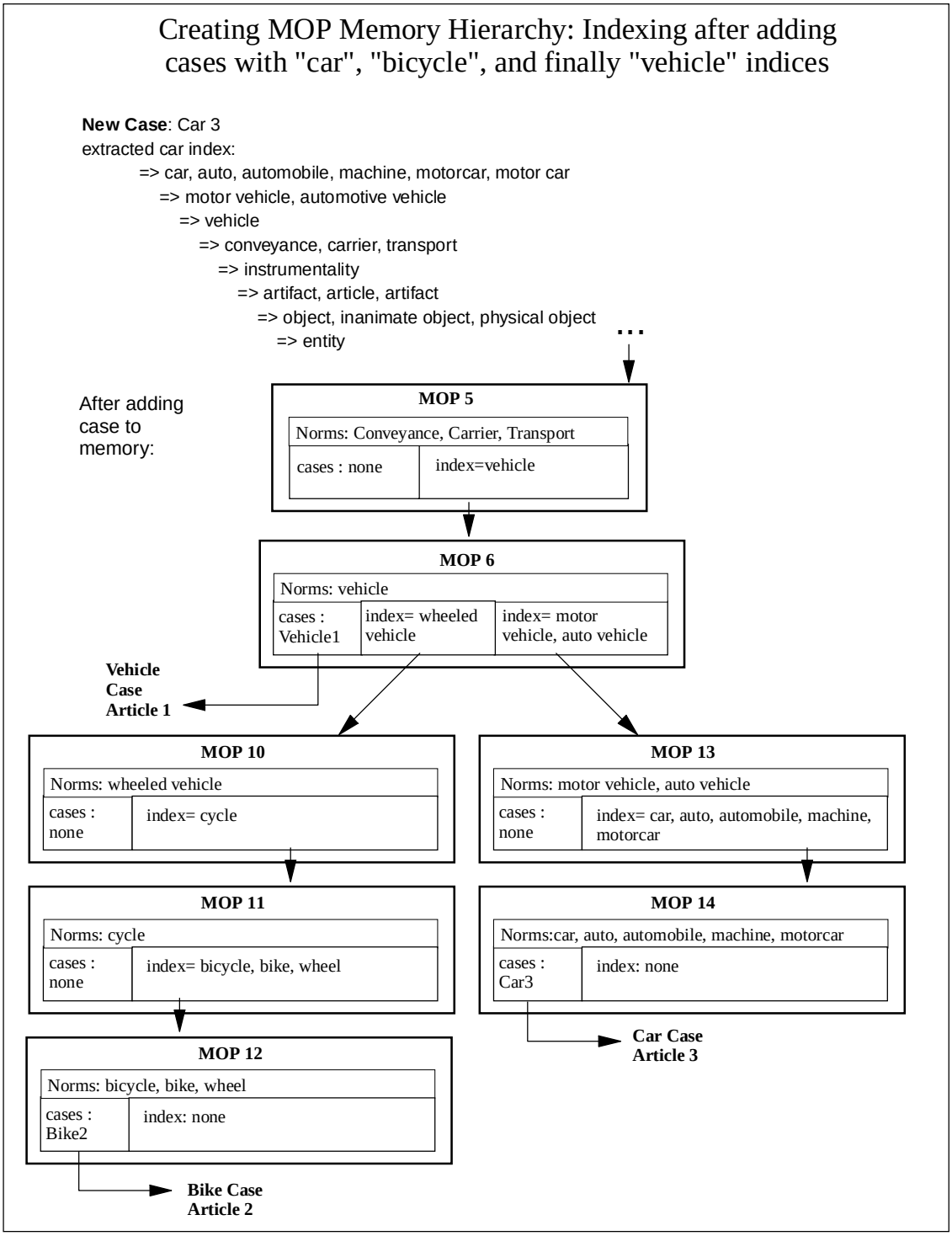


Figure 22: Adding third case to memory regarding car. The car case is added to the portion of the hierarchy matching the vehicle, then new specializations added to the bottom of the hierarchy. The car is indexed separately from the bicycle based upon their differences.

These simplified examples assume that each case is represented by only a single index, while in practice each case will be represented by many indices. Initially, the memory hierarchy is empty. Upon adding the first index regarding vehicles, the resulting memory hierarchy is shown in figure 20. Since there is nothing else in memory, the hierarchy is simply a linear list from the root to the case. Upon adding the second case regarding bicycles, this is merely a specialization of the vehicles index. Consequently, the hierarchy is still linear, except additional MOPs are added to represent the bicycle specialization as shown in figure 21. Finally, when the car case is added to memory as shown in figure 22, the case is indexed based upon its differences from other cases. The index differs in the type of vehicle, resulting in a fork within the memory hierarchy.

5.2.1 Matching Indices

Throughout the previous discussion, one detail has been glossed over: how are indices linking cases compared to each other? To traverse the hierarchy requires comparisons among these indices. If the indices match exactly, then these two are obviously computed as identical. However, some indices are comprised of partial overlap. For example, one index linking cases may be "car, motorcar, machine" while another index may be "machine, tool." Both indices share the term of "machine"; should this be enough to treat the indices as identical?

The approach taken in INFOS has been to consider indices to match if at least *WNMATCHPERCENT* of the terms in the query index are contained within the target index. This percentage was set to 85% to allow only closely matching indices to be traversed. However, to allow looser or even more restrictive matches, this value can be easily modified.

5.2.2 Generalization and Differences from CYRUS

While Kolodner's original CYRUS system performed a step known as *generalization*, this step is not necessary in INFOS since the indices are already organized hierarchically before insertion into memory. In CYRUS, the indices used were individual tokens; e.g. the simple index "transportation=car" instead of a complete hierarchy representing the meaning of car. Consequently, generalization was necessary in order to form abstractions from these flat concepts. This step can be bypassed in INFOS since the extracted indices already have structure.

Other differences from CYRUS include token-based indices in the memory hierarchy instead of attribute-value indices, and only partial indexing in the INFOS memory hierarchy as opposed to complete indexing in CYRUS. The attribute-value indices of CYRUS allow retrieval to proceed quickly by limiting traversal to those indices whose attributes exactly match the value. This has been replaced in INFOS with the partial matching scheme describe in the previous section. However, retrieval will still be quick in INFOS since search is also limited only to matching indices, not a complete search of the entire hierarchy.

An additional difference between the two systems is the amount of indexing; in CYRUS, all features are used to index each other. In other words, there is duplication among features; there is a path from the root to index A to index B to a case, as well as a path from the root to index B to index A to a case. This results in a rich hierarchy with many paths to an instance, but there is a large amount of redundant information. In INFOS, memory has been compacted so that an index hierarchy is not indexed from other indices but is indexed individually from the root. However, portions of indices are shared when they reference similar concepts, as in the example hierarchy shown for "bicycle" and "vehicle." This results in a much more compact memory hierarchy occupying less memory and system resources, but still allows cases to be retrieved.

5.2.3 Size Limitations

While using a case-base for articles that are read can be an effective means of classification, there are storage limitations to storing all messages ever read along with the memory hierarchy. Currently, INFOS assumes ample disk space is available to store the articles and the memory hierarchy. However, a number of options exist if storage space is low. One solution is to add a new case only if no other cases exist that are similar to the new case within a threshold value, *V*. The determination of a match is described in the memory retrieval section. Additionally, an aging process can be implemented that marks the recency of cases that are accessed; the oldest cases that have not been accessed can be removed by the user if desired.

The disk space required to store the articles increases linearly as articles are read. However, to examine the issue of required disk space to store the memory hierarchy, the size was examined as up to 425 messages were processed. The results are shown in figure 23. Initially, almost 1MB is required to store the initial indices as MOPs are created. However, as more indices are added, there is an increase in the amount of sharing among existing branches of the hierarchy, and less memory is required. Consequently, the memory required begins to level off asymptotically.

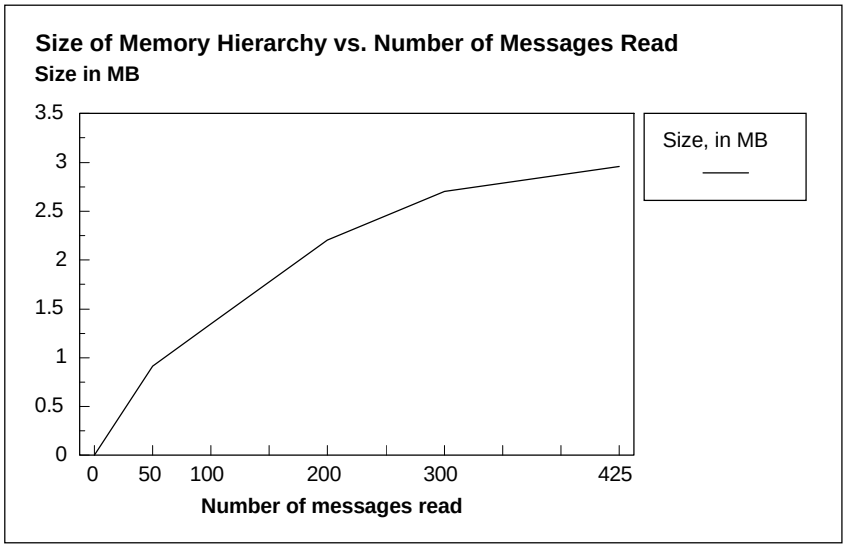


Figure 23 : Size of memory hierarchy as messages are indexed.

5.3 Memory Retrieval

Case-based memory retrieval involves searching for applicable cases based upon a given set of features. These features are simply WordNet sense indices from a new article that needs to be classified, and they are selected using the same algorithm described in section 5.1. Case retrieval in INFOS is fairly simple. A depth-first search is performed along indices that match the input query until cases are retrieved or there are no more input indices to follow. There are five cases that may occur while searching:

1. No indices from the query match in the memory hierarchy. This indicates the desired material is not located in memory, or elaboration strategies are required to find them.
2. The query matches exactly to a case in the memory hierarchy. This case should be returned as a complete match, indicated in INFOS by a match value of 1.
3. The query matches indices exactly in memory, but there is no case at the end of the hierarchy. For example, this occurs if the query is “vehicle” but memory only contains the definition for “bicycle.” A vehicle will completely match a bicycle since it is a

more general form of a bicycle. Consequently, the bicycle case is retrieved by retrieving all cases located below the query, but these cases are penalized by the amount they differ from the query. In this example, the penalty is the amount a vehicle differs from a bicycle. If the depth of the bicycle case in the hierarchy is *D*, and the depth of the vehicle query match is *Q*, then the partial match value returned is *Q/D*.

4. While performing the depth first search of the query among matching indices, a link to a case is found. For example, this will occur if the query is "bicycle" but memory only contains the definition for "vehicle." The search is not complete, since the query is more specific, but does match the definition for the more general vehicle. As in case 3, these articles are relevant and are returned, but they are penalized by the amount the query overshoots the case. If the depth of the bicycle query is *Q*, and the depth of the vehicle case in the hierarchy is *D*, then the partial match returned is *D/Q*.
5. The query only partially matches the links in memory. To allow for partial matches, non-matching links were allowed to be traversed as long as the total mismatch percentage was below *INDEXMATCHPERC*. This percentage is computed by counting the number of mismatched links, and dividing by the depth of the query. In INFOS, this percentage was set to 75% to allow some non-matching links to be explored, but to also cut off search if more than a handful of irrelevant links were examined. Any cases retrieved were penalized by subtracting the number of mismatched links *L* from the query depth *Q*, and multiplying the resulting match by *L/Q*. The *INDEXMATCHPERC* value can be increased to force closer matches only, or decreased to allow even more general matches.

As an example of case 5, consider the WordNet sense concept for automobile, which is represented as:

car, auto, automobile, machine, motorcar, motor car
=> motor vehicle, automotive vehicle
=> vehicle
=> conveyance, carrier, transport
=> instrumentality
=> artifact, article, artifact
=> object, inanimate object, physical object
=> entity

while the concept for bicycle is represented as:

bicycle, bike, wheel
=> cycle
=> wheeled vehicle
=> vehicle
=> conveyance, carrier, transport
=> instrumentality
=> artifact, article, artifact
=> object, inanimate object, physical object
=> entity

In comparing the auto query to the bicycle representation in the hierarchical memory, the bicycle query will match the representation for auto exactly down to the MOP that represents "vehicle." The next two sets of terms will not match, resulting in a match percentage of 6/8 or 75%. Consequently, the bicycle case will still be retrieved, since this match is still exceeding *INDEXMATCHPERC*, but the match return value is penalized by 25%.

While the process described above is fast, it may fail in finding a match at all if the wrong terms or keywords are supplied. In this case, the search algorithm may proceed by attempting to expand the search by modifying the search query. The process of searching for the right search queries is called elaboration. Kolodner implements a variety of elaboration strategies in the CYRUS system. Possible strategies include context-to-context and component-to-context elaboration strategies (infer a new context based on the current context or a component to search in an alternate branch of the hierarchy) or component-to-component elaboration strategies (infer a new component to search potentially deeper in the hierarchy). In this project, elaboration strategies were not

necessary since queries always retrieved case articles. However, if the straightforward retrieval process does not retrieve any cases, then elaboration can proceed by searching upon related part-of terms that may be generated from WordNet.

The search algorithm outlined above allows matches to be made between exact matches, specific queries and general memory objects, general queries and specific memory objects, and objects that are similar, but slightly different. For each case that is retrieved, the overall match value for that case is computed by summing over all *n* feature queries the following distance function:

$$Match = \frac{1}{n}\sum_{i=1}^{n}(MatchPercent_i) \times \mathrm{Re}\,levancy_i \qquad (5)$$

In equation 5, the relevancy of a term is computed according to equation 4 (section 5.1) given in the feature extraction section. The relevancy for terms present in the input but not in the retrieved case is 0, and the relevancy for keywords indexed through the inverted index is 1. The MatchPercent value for these terms is computed through equation 1 (section 4.3), the same method used for keywords in the global hill climbing scheme.

In INFOS, the retrieved cases are sorted by degree of match. The classification statistics of the best matching case can then be used to classify the new article, using the Accepted and Rejected counters for the case and computing a classification via equation 2 (section 4.3). The article number can also be displayed as a justification to the user to indicate why INFOS believes new articles should be classified similarly. Since the case base marks which sentences relate to specific topics through the topic stem identification step, exact sentences can be displayed to the user as a justification for why the new article is classified the way it is. Furthermore, in the event that a user is searching for previously read concepts, the exact sentences can be quickly displayed to the user.

Another possible classification technique is to use the complete set of retrieved articles for classification, rather than single out one case. In this method, proposed by

Riloff, the set of retrieved cases is statistically averaged together to result in a classification of the new article (Riloff, 1993). The classification can then be performed through equation 2 (section 4.3) using the combined Rejection and Accepted counters of all retrieved cases. Both of these classification methods were tested and experimental results are given in section 5.5.

5.4 Case-Based Scheme for Information Retrieval

To examine the effectiveness of the case-based system with existing information retrieval systems, the case-base was trained and tested upon the Time test set[1]. This set has been popular for use in the information retrieval community and consists of 425 articles extracted from Time Magazine in 1963. Along with the documents are 83 queries. Each query is one to three sentences long and describes one or more articles from the test collection. A set of relevancy judgments, provided by experts, are also included that indicate which of the 425 articles corresponds to the query. Each query was typically related to anywhere from 1-7 documents.

When processing the 425 articles, INFOS only trained up to the first 20 sentences of each article. Some articles contained more than 20 sentences; this additional text was discarded in order to expedite the test. This resulted in the failure to retrieve some queries that were related to text found at the end of long articles. After the articles were processed, INFOS was given the queries and a list of retrieved articles produced and sorted in rank through the case-based process. Since INFOS retrieves a large number of articles while a query may only have a handful of relevant articles listed, a hit for query *Q* was defined as occurring if an article was listed within the top *X* entries of INFO's list of returned items for *Q*. *X* is defined as the 3 times the number of documents actually listed

[1]This and other test collections are available at
http://www.dcs.gla.ac.uk/idom/ir_resources/test_collections/lisa/

as being relevant to query *Q* in the relevancy judgment file. This credit system allows INFOS the flexibility to retrieve articles and force the system to sort the truly relevant files near the top, but not necessarily at the very top.

The results using INFOS for case-based retrieval alone and when combined with the inverted index scheme are shown in table 7. Additionally, results are also shown for the tf-idf method. The tf-idf method was also only trained upon text from the first 20 sentences of each article.

Method	**Documents Correctly Retrieved**	**Documents Missed**	**Precision**
Case-Based Alone	215	111	66%
tf-idf	247	79	76%
Case Based + Inverted Index	288	38	88%

Table 7: Results of Information Retrieval upon Time dataset

The table shows that the CBR method alone performs the worst of all the techniques, retrieving only 66% of the articles. The low performance occurs primarily when topic senses are inadequately extracted from the text. As a result, the article is indexed with irrelevant definitions; for example, the burning sense of the word “fire” instead of the shooting definition of fire in a sentence such as “He fired the rifle.” The problem of determining the proper word sense disambiguation is an extremely difficult one, and the results indicate the index extraction performs only marginally. Expansion of words into their definitions can be detrimental unless the correct sense is known, because adding the relatives of the wrong sense adds additional noise to the matching process (Voorhees, 1994). Nevertheless, despite this problem, the CBR method still performed close to the tf-idf method.

The true benefit of the CBR method occurs when it is combined with the files retrieved through the inverted indices. Recall that the inverted index system used in

INFOS is a slightly simpler version of tf-idf since the term frequency in all documents is ignored. When the CBR classifications were combined with the files retrieved through the inverted indices, precision of 88% was achieved, higher than tf-idf alone. This indicates that there is merit in using the CBR system. The process of conceptual information retrieval does retrieve articles in which the keyword schemes have difficulties. These results also indicate that the best uses for the CBR scheme may be achieved when combined with keyword approaches.

5.5 Results of Case-Based Scheme

Experiments were also performed using the case-based system on the Usenet news domain. These tests were performed using both the best matching case classification and the average class classification scheme discussed in section 5.3. Additionally, the same tests that were run through the global hill climbing scheme were also run with the case-based scheme. Finally, the case-based scheme was tested when used in conjunction with the global scheme. When used in conjunction with the global scheme, the global scheme classification was performed first. If the global scheme returned an unknown classification, then the classification of the case-based scheme was used. The global scheme was performed first because it was found to have a lower error rate, and is also quicker than the CBR method. If a simple method can provide the correct classification, it is reasonable to use that method before more complex ones are attempted. The overall results indicate that the semantic and keyword based filtering provided by the combined best match CBR and global hill climbing scheme performs best. The hybrid method improved the correct classification percentage over the global hill climbing method alone.

5.5.1 Experimental Results - Consecutively Posted Articles

Using the same data provided by subjects from the experiment described in the global hill climbing experiment (chapter 4), INFOS with the CBR component was trained upon the same 50 articles selected at random from the set of 100 consecutively posted messages. Classifications were made and performance evaluated according to the methods presented in chapter 4. A summary of the results is shown in table 8 showing the Global Hill Climbing method (GHC), case-based reasoning alone using the average of all retrieved cases to make a classification, case-based reasoning alone using the classification of the most highly ranked case as the classification, and both CBR schemes combined with the global hill climbing method. The combined CBR scheme using best match with global hill climbing performed best, although it had a slightly higher error rate than the global hill climbing method alone.

Classification Method	**Percentage Classified Correctly**	**Percentage Classified Unknown**	**Percentage Classified Incorrectly**	**Percent False Positives**	**Percent False Negatives**
Global Hill Climbing (GHC)	51.5	40.9	7.3	50	50
CBR - Averaging across Cases	54.2	26.8	18.9	45	55
CBR - Best Match as Class	39.8	50.5	9.5	77	33
Combine GHC + CBR Average	62.1	20.9	16.8	47	54
Combine GHC + CBR Best Match	58.0	29.9	12.1	62	37

Table 8: Classification accuracy for various methods.
Results are averaged over 14 subjects, showing percentage classified correctly, incorrectly, and unknown for 100 consecutively posted articles, 50 articles read. Methods tested include Case-Based Reasoning (CBR) method using average of retrieved cases, CBR using best matching case as classifier, and combined versions of both with Global Hill Climbing (GHC) method. The types of errors in both schemes are also reported.

The results from this experiment indicate that the global hill climbing method still has the lowest error. As described in section 5.4, the case-based scheme will have some poor indices due to the sense disambiguation problem that can allow irrelevant cases to be retrieved. Consequently, the CBR schemes both have higher error rates than the global hill climbing method. The sense disambiguation problem may also account for why the CBR scheme that averaged all retrieved cases returned the highest error rate; irrelevant cases were factored into the classification. However, the best match CBR method returned error rates only slightly higher than the global hill climbing scheme, probably because the best matching case is more relevant to the input article. When combined with the global hill climbing scheme, the best match CBR method does achieve a slightly higher correct classification percentage at 58%, and a slightly higher error rate at 12%. However, this error was comprised primarily from false positives, a less serious error than false negatives. One explanation for this phenomenon may be a prototype article representative of an entire thread that tends to be retrieved and ranked first for all articles in that thread; the other schemes may average out the effects of individual articles, resulting in more even balance between false positives and false negatives.

5.5.2 Experimental Results - New Threads - Unread Articles

Experiments were also conducted using the same classification methods upon the dataset with completely new article threads. This simulates a reader who has built up a user profile, but has not read any articles for several days or weeks, resulting in new article threads and topics. The combined weights for the global hill climbing scheme were identical to those described in section 4.8, and the experimental methods were also identical. The results are shown in table 9. The combined global hill climbing and best match CBR algorithm produces the best results, indicating that the CBR scheme and conceptual indexing is useful when addressing a new set of data.

Classification Method	Percentage Classified Correctly	Percentage Classified Unknown	Percentage Classified Incorrectly	Percent False Positives	Percent False Negatives
Global Hill Climbing (GHC)	30.9	59.8	7.7	71	29
CBR - Averaging across Cases	53.0	28.0	18.9	47	53
CBR - Best Match as Class	25.7	69.5	5.3	54	46
Combine GHC + CBR Average	59.8	16.6	23.5	30	70
Combine GHC + CBR Best Match	43.9	43.9	12.1	65	35

Table 9: Classification accuracy for various methods.
Results are averaged over 14 subjects, showing percentage classified correctly, incorrectly, and unknown for 50 articles from new threads, 100 articles read. Methods tested include Case-Based Reasoning (CBR) method using average of retrieved cases, CBR using best matching case as classifier, and combined versions of both with Global Hill Climbing (GHC) method. The types of errors in both schemes are also reported.

Once again, the best match CBR scheme performed best overall. The CBR averaging scheme did result in a higher percentage of correct classifications, but also had a much higher error rate. In this experiment, the CBR scheme significantly improved upon the performance of the global hill climbing scheme alone, increasing an additional 13% while contributing 4% additional error. The reason for the increase in performance is that semantic content is more important when a set of completely new threads and articles is involved. This new set of articles will have fewer keywords in common than the old set of data, lowering performance of keyword methods. However, by mapping the new vocabulary into concepts, correlations can be made with previously read articles. These concepts are identifiable by the case-based scheme, but not through the keyword scheme.

5.6 Long Term Studies

A longer term study of INFOS's CBR best match/hill climbing scheme was conducted using three subjects reading the ucd.life newsgroup. The subjects read a total of 400 consecutively posted messages. In order to simulate how users actually read messages, the 400 messages were broken up into six groups of 50 articles: articles 1-50, 51-100, 101-150, up to 351-400. Subjects read and classified all 400 articles. Starting with the first group of articles, INFOS was trained upon 20 articles randomly selected from each group and then classified the group's unread articles along with unread articles from earlier groups. This procedure is similar to how users actually read messages: a new batch of articles appears, users read a selection of these articles, and then INFOS classifies the rest of the articles and future articles for users to peruse. Averages for the three users indicating the percentage of articles classified correctly, incorrectly, and unknown on this data using the CBR/Global Hill Climbing method is shown in figure 24.

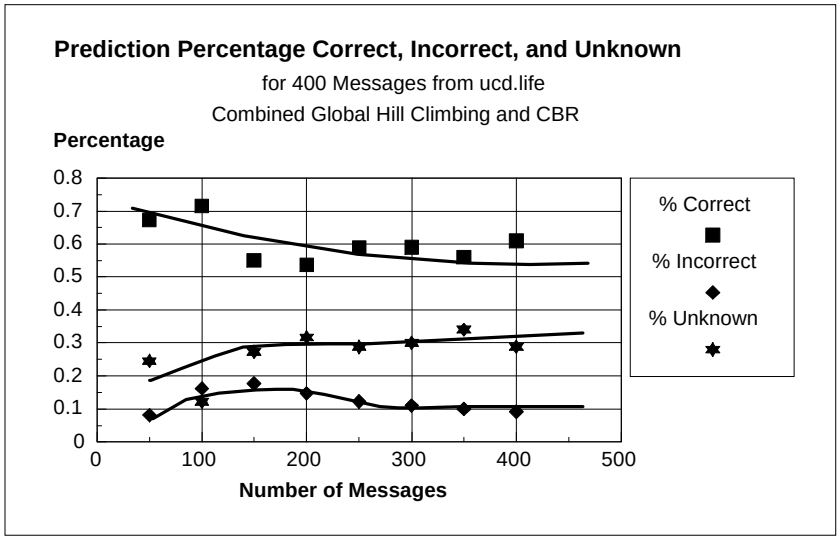


Figure 24 : Long Term study for Combined CBR/Global Hill Climbing.
Subjects read 400 articles in groups of 50, training was performed upon 20 randomly selected articles from each group, and testing was performed on the unread articles of the group. Performance stabilizes at 60% correct, 30% unknown, and 10% incorrect.

Initially, the user model is empty and the results are slightly erratic due to the lack of knowledge in the user model and the lack of diverse topics in some of the data sets. However, after approximately four of the message groups have been processed and a larger diversity of messages encountered, the classification rates stabilize at close to 60% correct, 30% unknown, and 10% incorrect. These percentages are very close to those obtained in the previous experiment and described in Table 8. Note that the percentage of correct messages stabilizes at approximately 60% instead of continuing to rise as the user model becomes more accurate. Factors that prevent INFOS from classifying more articles correctly include an influx of new topics that INFOS has not yet learned to classify and changing user interests.

5.7 Chapter Summary

This chapter has described the case-based reasoning component of INFOS along with its integration with the global hill climbing method. The highlights of this scheme include:

- Use of WordNet to map words into a conceptual abstraction hierarchy. Words and phrases can now be compared at a conceptual level.

- Modified version of Paice's algorithm to extract candidate nouns and verbs for WordNet lookup and subsequent indexing based upon the concept of topic neighborhoods.

- Cases are indexed in memory through a modified version of Kolodner's memory creation algorithm. Cases are indexed in an abstraction hierarchy, facilitating quick retrieval and the automatic clustering of cases by similarity.

- Retrieval of cases is executed by performing a depth-first search through the memory hierarchy along matching indices. By following mismatches, partial matches can be retrieved (e.g., a bicycle query may retrieve cases regarding automobiles).

- The case-based scheme may be used for information retrieval. When coupled with an inverted index, the CBR scheme outperformed tf-idf on the Time dataset.

- When the global hill climbing scheme is invoked first, and then the case-based reasoning method invoked when the global hill climbing scheme fails, the accuracy rate increased to 58% although the error rate also increased to 12%. The increased error is due to inaccurately disambiguated nouns and verbs. When used over a longer period of time, the accuracy rate stabilized near 60% and the error rate stabilized near 10%.

6. Genetic Algorithm Method

While a case-based reasoning system can successfully augment a keyword based approach with semantic knowledge and improve filtering recall, a genetic algorithm can provide for exploration and increased power. In NewT, Sheth showed that a genetic algorithm helped readers explore other newsgroups by applying user models across different newsgroups (Sheth, 1994). These readers found interesting articles in newsgroups they would not normally read. This approach could certainly be applied to INFOS, since each newsgroup contains its own global hill climbing table. However, can the genetic algorithm also be applied to within a single newsgroup to aid in exploration? A genetic scheme was constructed to address this question in INFOS. This scheme is called the *local genetic hill climbing method*, since it augments the hill climbing scheme by using hill climbing to learn in addition to learning through genetic algorithms. Experimental results indicate that the genetic scheme, when coupled with the hill climbing algorithm, can produce better results than the global hill climbing scheme alone by exploring the news space and performing non-linear classifications among article features.

6.1 Local Genetic Hill Climbing

The local genetic hill climbing method examined in INFOS is similar to the global hill-climbing method except instead of a single dynamic table there is a population of many tables, where each table constitutes an individual, and each individual performs its own hill climbing as well as genetic crossover (Holland, 1975; Mock 1992). Each individual in the population can be viewed as an agent attempting to model some aspect of the user's interests. Those individuals that do well at modeling the user will thrive and survive,

producing similar offspring. Those individuals that do a poor job will die. Eventually the population will adapt so individuals accurately model the user.

Additionally, through a collection of tables, the limitations of the global hill climbing scheme's linear classification are removed. In the global hill climbing method, a word can have only one definition. No matter what context a word appears, the same table weights will be used. With a genetic method using multiple tables, the same word can appear in separate tables but with different weights; in this manner, separate tables can serve to identify different meanings of a word if the weights are updated differently. However, for this method to be useful, other words that indicate the context of the ambiguous word will have to be separated in each table so that the correct table is used. For example, to disambiguate the word "shark" one table could contain weights for the words "great" and "white" to indicate the fish, while another table could contain weights for the words "San Jose" to indicate the hockey team.

An introduction to genetic algorithms is given in section 3.4. Additional background information regarding the operation of genetic algorithms can be found in Goldberg (1989). To run the genetic algorithm, the table size was set to 20, allowing each individual to identify combinations up to 20 features. Most articles could be uniquely identified through 4 or 5 features; consequently, each individual has the capacity to represent at least 5 specific messages and perhaps more general types of messages. These feature tables comprise the chromosome of each individual. The entries in each table are initialized to features selected randomly from a set of 100 previously stored messages, and the accepted and rejected values initialized to small random numbers between 0 and 5. In addition to the table, each chromosome also consists of a history variable that counts the number of times a particular individual has made a correct, or incorrect, prediction. This variable is initialized to zero.

6.1.1 Hill Climbing Component

The genetic algorithm classification procedure is similar to the global hill climbing method, except there are now many tables to examine instead of one global table. Since the best-match approach worked best for the case-based scheme, the classification approach for the genetic scheme first finds the single individual that best matches the input. This is determined by computing how many features from the query input are present in the individual. The individual with the most matching features is selected as most representative of the input article. If at least *C* percent of the features match, then the accepted and rejected values from the matching features of this single individual are added and used to compute a prediction in a manner identical to that of the global hill climbing approach. If the best individual doesn't match at least *C* percent, it is deemed too distant from the input to make an accurate prediction, and the prediction is set to Unknown. In this project, *C* was set to 50% so at least half of the features must match to activate an individual. The hill-climbing learning procedure is outlined next. Each new article read by the user is classified as accepted or rejected. If rejected, then the rejected values of all features matching the input are incremented in the tables of all individuals. Similarly, if accepted, then the accepted values of all features matching the input are incremented in the tables of all individuals. This is a global update similar to the global hill climbing approach.

The next step is to update the individuals with greater than *C* percent of its features matching the features of the message. The prediction of all of these individuals is computed; if the prediction equals the actual value, then the history variable is incremented. If it is incorrect, the history is decremented. This history variable stores the number of times a particular individual has been correct, and will be used in the genetic algorithm fitness function.

The final step is to create new individuals if necessary. If the best matching individual has a match percentage less than *C*, then no individual exists who matches the input and one must be created. The individual with the smallest fitness and lowest match percentage is selected and its features set to those of the input. In addition to exactly matching new messages that are far away from existing individuals, this process also adds new features into the gene pool for the genetic algorithm component of the system.

6.1.2 Genetic Algorithm Component

After the user finishes reading messages, the genetic algorithm component goes into effect. Rather than store all previously read messages for training, the system trains in batches, using only the set of messages read in the previous session. However, if desired, training could be performed on all messages since the CBR scheme will save messages anyway. This approach was not taken in these simulations due to the increase in training time that would result. With a fitness function based upon these messages alone, it is possible that individuals who classified well on old messages would classify poorly on the new message set. Consequently, it is possible that potentially good individuals that performed well on old messages will be given an unfair fitness. These individuals may then be left out of mating, or may be destroyed during the crossover process.

To help make the fitness more fair to these old individuals, the history variable is included into the fitness calculations. This will bias the fitness by including the previous performance of each individual in the past. For each message in the set of articles read, a prediction is computed using the equation 3 (section 4.4). The fitness of each individual is given by the history variable plus the sum of the correct predictions minus the sum of the incorrect predictions. The crossover operation uses Fitness Proportionate Reproduction (Holland, 1975) to select two parent individuals from the population. The actual crossover operation is two point crossover; two indices are selected at random from one

of the parents, and all features and values within this selection are switched among parents. The history variables for both parents are then set to the average of the history value of the two parents. The mutation operation works by selecting a single individual at random and then either adding a positive or negative random value to one of the feature values, or by selecting a new feature at random from the set of new messages and randomly replace an old feature.

6.2 Local Genetic Hill Climbing - Experimental Results

The global hill climbing algorithm and the local hill climbing genetic algorithm were both tested on a set of 100 messages extracted from the **comp.ai** artificial intelligence newsgroup. Three users read all of these messages and marked each as accepted or rejected. The first 50 messages were used for training, and the system predicted the users' choices for the rest of the unread messages and averaged the correct, unknown, and incorrect predictions. These results did not incorporate collaborative data that would be collected when multiple users share reviews with each other.

For the genetic scheme, the number of generations varied from 0 to 5, the probability of mutation was set to 2%, and the population size was set to 50. With the exception of the number of generations, these are fairly standard settings (Goldberg, 1989). The number of generations was kept low, since Sheth showed that a large number of generations results in excessive mixing of user models. In his experiments, only a small number of generations were run, compared to traditional genetic algorithms (Sheth, 1994).

A summary of the results for both methods is shown in table 10. In this experiment the global method had the lowest error rate at 2% and best classification percentage at 52%.

Classification Method	Percentage Classified Correctly	Percentage Classified Unknown	Percentage Classified Incorrectly
Global Hill Climbing	52	46	2
Genetic, 0 generations	42	50	8
Genetic, 1 generations	44	52	4
Genetic, 3 generations	42	54	4
Genetic, 5 generations	36	58	6

Table 10: Classification on comp.ai articles for Global and Genetic Schemes

The percent of correct classifications for the best genetic method of 1 generation is slightly lower than the global hill climbing scheme at 44% opposed to 52%, probably due to the slower training time than the global scheme, and the relatively small amount of training performed (only 50 messages). The results are similar when run with 3 generations per batch, but accuracy decreased with 5 generations. At this point, the amount of crossover destroyed too many old individuals necessary to perform well, and the accuracy of correct predictions decreased down to 36%.

While these results indicate that the global scheme performs better on this set of data, further testing is necessary before conclusions may be drawn and generalized. In particular, a larger data set and testing with larger batch sizes may result in significantly different performance. Furthermore, the genetic scheme does perform okay on its own, correctly classifying a large portion of the data set and making relatively few errors.

Further insight may also be gained by examining which messages are being classified by the genetic scheme and comparing these to the answers classified by the global scheme. Comparison of these answers indicates that the genetic scheme often makes correct classifications of messages that the global scheme marked as unknown. In other words, the genetic scheme has effectively enlarged the search space to include new areas that the global scheme would not consider. In the run with 1 generation per batch, 18% of the correctly classified answers were classified as unknown by the global scheme. This percentage reached 30% in the run with 3 generations per batch. Since the genetic

scheme is capable of classification in cases where the global scheme is not, the two systems can be combined together to produce a single system with higher overall precision and breadth than either alone, just as was done with the global hill climbing method and the CBR method.

Table 11 shows the results when messages classified as unknown by the global hill climbing method are then classified using the genetic scheme. Performance is improved significantly, up to 64% correct classification with low error.

Classification Method	**Percentage Classified Correctly**	**Percentage Classified Unknown**	**Percentage Classified Incorrectly**
Global Hill Climbing	52	46	2
Combined Method , 1 generations	59	38	3
Combined Method, 3 generations	64	32	4

Table 11: Combined Genetic + Global Hill Climbing scheme correctly classifies messages unknown by Global Hill Climbing Method for improved overall performance

The results from table 11 indicate that a few generations of the more powerful genetic algorithm scheme can help users explore the search space and increase classification precision. Note that this scheme has only been applied to the individuals operating within the same newsgroup. Future work is required to investigate the effects of combining individuals operating upon different newsgroups.

6.3 Chapter Summary

This chapter has described the implementation of the genetic algorithm component of INFOS. The highlights of this method include:

- Local genetic hill climbing applies a genetic algorithm to a population of global hill climbing tables. This method is capable of performing non-linear classifications due to the separation of tables, but is more difficult to train than the global hill climbing method.

- When used alone, the genetic algorithm scheme performed worse than the global hill climbing method. However, when combined with the global hill climbing scheme, classification accuracy increased by 10% over the global hill climbing method while the error rate increased by only 1%.

- Experimental results indicate that genetic algorithms are useful for exploring other areas of the news space, and provide for a controlled method of diversification. This may be useful to prevent a news filter from defining a user's interests too narrowly.

7. From Word Recognition to Phrase and Sentence Recognition

The indexing methods employed in the global hill climbing, case-based reasoning, and genetic algorithm chapters are all based on *bottom-up* knowledge. Given the meaning of individual words, statistics or cases are referenced that may be relevant. In addition to bottom-up recognition, *top-down* methods are also employed by humans. Cases may be inferred or referenced through scripts, plans, or other conceptual structures. These are all high-level knowledge structures. However, before they can be created, concepts at the word, phrase, sentence, and paragraph levels must all be parsed and recognized. This chapter investigates the identification of concepts at the level of phrases and sentences.

7.1 Index Patterns

WordNet identifies concepts at the level of words and common phrases. However, the definition of words in isolation does not always correctly index an article. If we are interested in games that the Sacramento Kings basketball team has won, then appropriate concepts for search and retrieval include “Sacramento Kings”, “basketball”, and “win.” However, retrieval upon these concepts (or keywords) will also match articles in which the Sacramento Kings lost, but the opponent has won. The concepts of “Sacramento Kings”, “basketball”, and “win” are all present in these incorrectly retrieved articles, except the matching “win” is in reference to the opponent, not the Kings. The knowledge required to distinguish the winner is the syntax that pieces together concepts to form sentences and phrases.

Identifying concepts at the level of phrases and sentences through syntax and semantics is the major thrust of parsers. Although the parsing process is simplified in INFOS, the parsing method employed is based upon case-based parsing (Riesbeck and

Martin, 1986). Case-based parsing is seen as a recognition process requiring memory organization and search. As described in the previous section, MOP structures are used to represent concepts. These concepts include linguistic, domain, and world knowledge. A MOP may be used to represent a word, a concept, or any other semantically useful item within memory. As proposed by Riesbeck and Martin, there are MOPs for each word that the system knows, as well as MOPs called *index patterns* (Riesbeck & Martin, 1986; Riesbeck & Schank, 1989) that represent stereotypical mappings of natural language into their corresponding concepts. These index patterns drive the parsing process. Other MOPs represent processing knowledge the system has to aid in constructing case descriptions.

The actual parsing of a text involves the recognition of slot-filler relationships between concepts in a given text. As an example, the FANSYS system (Alvarado et. al., 1993; Alvarado & Mock, 1995) employs index patterns to parse cases of powering an object on. The index pattern *{ <actor> "powers" "up" <object> } => M-POWER-UP-EVENT* specifies that a power-up event can be recognized if an actor concept followed immediately by the words "powers up" followed by some object concept is recognized from the input text. The actor and object concepts are slots in the power-up event representation; the index pattern represents the linguistic or conceptual relationship between those slots as expressed in English. Patterns are associated with the MOP that they reference and the MOP with which they are stored by lexical links.

These index patterns also specify lexical and (implicitly) semantic constraints. In the example of the powering-up event, the words describing the actor MOP are required to be adjacent to the words "powers up" in order for the pattern to be recognized. Patterns thus implicitly encode syntactic features of a target language. Concepts specified within an index pattern are further constrained by filler restrictions specified within the MOP to which the slot pertains. In the powering-up example, the object in the pattern is constrained to be some mechanical device (e.g., a system) capable of being powered up.

7.2 Implementation of Index Patterns in INFOS

INFOS uses a simplified version of the MOP-based parsing process implemented in systems such as FANSYS. Rather than parse into instances of MOP knowledge structures to identify cases, the process implemented in INFOS is only concerned with linking from index patterns to a classification. For future work, INFOS should use index patterns to identify cases and then use those cases to determine a classification. However, for purposes of an initial investigation, INFOS currently associates a classification rating (suggested, not suggested) with each index pattern.

The index patterns implemented in INFOS are similar to those implemented in FANSYS, except they are not overlaid on top of a MOP hierarchy. However, index patterns are indexed in WordNet. The index patterns consist of either conceptual items or static lexical items. In order for a match to be made between a sentence and a static lexical item, the static lexical item must appear verbatim in the sentence. For example, if an index pattern consists of the static lexical entry *"up"* then the word "up" must appear in the text to match this entry. Additionally, all other entries in the index pattern must match in the same order as the target text for an entire index pattern to activate. The ordering constraint of index patterns preserves syntactical constructs present in the English language. Note that lexical entries need not be words; they may also be punctuation or other symbols.

In addition to the static lexical items, index patterns also consist of conceptual items. Conceptual items are simply concepts or abstractions. A concept from a text article matches a conceptual item from an index pattern if the text article's concept falls under the same category as the index pattern's concept. The categorical match is positive if at least one sense of the index pattern's concept is an abstraction of at least one sense of

the text article's concept. As an example, consider the WordNet concepts for "actor" and "mechanic" defined below:

actor, histrion, player, thespian, role player
=> performer, performing artist
=> entertainer
=> person, individual, someone, mortal, human, soul
=> life form, organism, being, living thing
=> entity
=> causal agent, cause, causal agency
=> entity

machinist, mechanic, shop mechanic
=> craftsman, artisan, journeyman, artificer
=> skilled worker, trained worker
=> worker
=> person, individual, someone, mortal, human, soul
=> life form, organism, being, living thing
=> entity
=> causal agent, cause, causal agency
=> entity

Both WordNet concepts may be represented hierarchically as shown in figure 25:

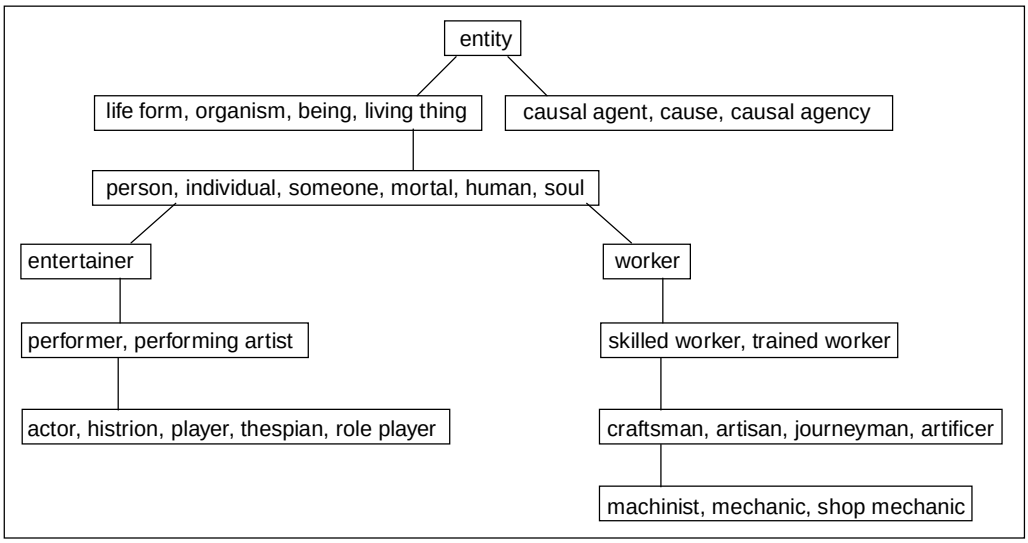


Figure 25: Partial WordNet Memory Hierarchy for "actor" and "mechanic"

In figure 25, if the concept for <*person*> is a conceptual item in an index pattern, then any concept below the node representing person will match <*person*>. For example, text containing the words "actor", "mechanic", "worker", or "performer" are all

considered matches with <*person*> since the node representing <*person*> is an abstraction of the nodes representing all of these words. However, text containing the words "life form" is not considered a match with <*person*> since life form's node is an abstraction of <*person*>, not vice versa. Similarly, the index pattern containing the conceptual item <*entertainer*> matches text containing the words "performer" or "artist" since <*entertainer*> is an abstraction of these concepts, but does not match text containing the word "mechanic" since the mechanic node is a cousin of <*entertainer*>, not a child or specialization.

By defining index patterns with conceptual items and static lexical items, index patterns can be created that match both broad concepts and exact lexical tokens. Additionally, the order in which concepts must appear restricts matches to specific syntactic structures. The syntactic restriction also supports disambiguation of word senses and lowers the probability of false matches for articles. The probability of a single incorrectly disambiguated word sense may be high and result in the retrieval of an irrelevant case. However, the probability of incorrectly disambiguated word senses appearing in the same sequence as an index pattern is low. Consequently, very few index patterns will be incorrectly activated and retrieval precision via index patterns will be high.

Despite the power of index patterns, care must be taken so that they are constructed correctly. A haphazard user may enter the index pattern *{ <actor> "powers" "up" <object> }* when he really intends to enter the index pattern *{ <person> "powers" "up" <object> }*. While these patterns may be accepted in colloquial English to represent the event of a person powering an object on, the difference between an actor and a person in WordNet is significant. In WordNet, the concept for actor is specified to the level of a role-player or thespian, while the concept for person is a much higher abstraction representing all types of people. Consequently, unless the creator of the index pattern wishes to restrict the actor to theatrical characters, the former pattern is incorrect

and a pattern closer to the latter should be employed[2]. Since care and knowledge of WordNet is necessary to create proper index patterns, the use of index patterns for a general population may be limited. However, patterns are not difficult to create once the WordNet hierarchy is understood. Understanding WordNet will be facilitated by improved methods to visualize and browse the WordNet hierarchy. Additionally, AI techniques that model user's intentions and reference index patterns against commonsense knowledge can help make index pattern creation more practical for use by a layman.

Figure 26 depicts the mapping of input text into the index pattern *{ <person> <search> <help> }*. If the user is not interested in reading articles where authors are asking for help about a particular problem then this pattern could be associated with a rejected classification. In this example, the article text contains the sentence "my employer is looking for assistance..." Since "employer" is a specialization of *<person>*, "looking" is a specialization of *<search>*, "assistance" is a specialization of *<help>*, and all of these components match in the same order specified in the index pattern, then the index pattern is activated and used to classify the text.

[2]"Actor" actually contains two senses in wordnet, one as the theatrical performer and the other as a type of worker. However, both senses are specializations indexed under the concept for "person."

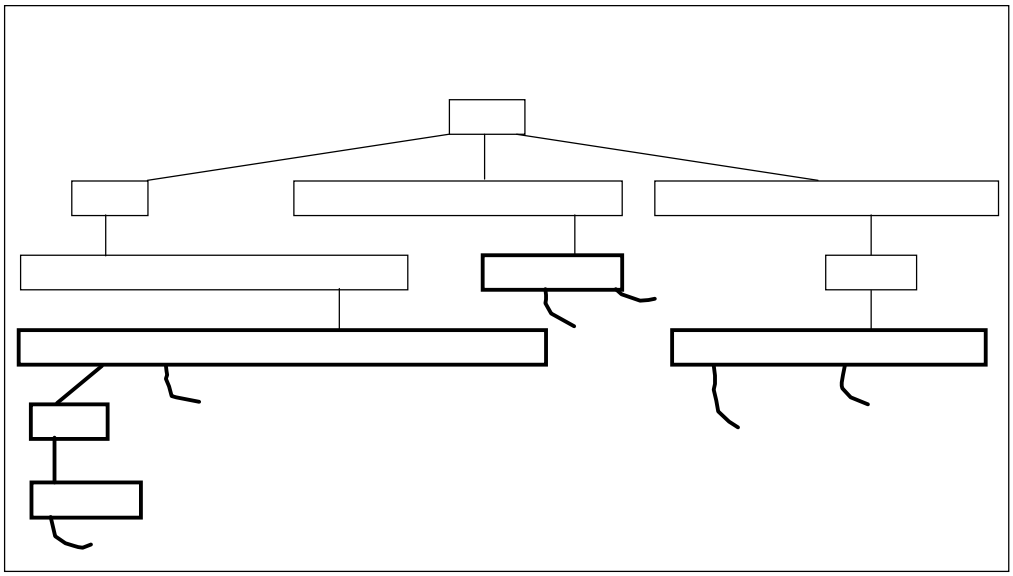


Figure 26: Activation of Index Pattern.
The pattern *{ <person> <search> <help> }* is activated by the text "employer is looking for assistance" since "employer" is indexed under <person>, "looking" is indexed under <search> and "assistance" is indexed under <help> in the same sequence. Each index pattern is also associated with a classification. The classifications of activated patterns are used to classify input text articles.

7.3 Experimental Results - Information Filtering via Index Patterns

Due to time constraints and the requirement that users be knowledgeable about WordNet's structure, index patterns were not implemented in the version of INFOS tested by users. However, a preliminary experiment was conducted to examine the effectiveness of index patterns for information filtering. In this study, 47 consecutively posted articles from the comp.ai newsgroup were selected. Specific concepts were selected as targets for positive retrieval (articles the user wishes to see) and others selected as targets for negative retrieval (articles the user does not wish to see). All 47 articles were read and classified by the author. The author-judged classifications served as the correct classifications that the system attempted to match.

The theme selected for filtering was the concept of an actor getting some type of object. Two concepts were selected for positive retrieval, the act of retrieving or finding

information and the act of getting some type of system. The index patterns implemented to represent these concepts are:

{ <person> <get> <information> => suggested}

{ <person> <search> <information> => suggested}

{ <person> <get> <system> => suggested}

{ <person> <search> <system> => suggested}

The concepts selected for negative retrieval, or filtering, included the concepts of searching for a location, searching for a person, and searching for a message (such as help). The index patterns implemented to represent these concepts are:

{ <person> <get> <location> => not suggested}

{ <person> <search> <location> => not suggested}

{ <person> <get> <person> => not suggested}

{ <person> <search> <person> => not suggested}

{ <person> <get> <communication> => not suggested}

{ <person> <search> <communication> => not suggested}

To perform filtering with index patterns, all articles were initially given a weight of 0. Each time a positive index pattern was triggered, the respective article's weight was incremented. Similarly, each time a negative index pattern was triggered, the respective article's weight was decremented. This scheme combines the index patterns in a simple, linear manner. In the end, those articles with a positive weight were classified positively, and those with a negative weight classified negatively.

As a benchmark, filtering was also performed using keywords, concepts alone, and a hybrid keyword/concept-alone method. The concept-alone method is identical to the index pattern scheme, except a pattern was considered to match with a sentence from article text as long as all the concepts in the pattern matched *somewhere* with concepts in the article's sentence. In other words, syntax and pattern order was ignored, as in the case-based scheme.

The keyword method attached an ambivalent weight to the words “get search look find”, a positive weight to the words “information system knowledge program work paper research”, and a negative weight to the words “location direction person communication message help assistance aid.” Articles containing substrings of these keywords were linearly updated with the respective weight value. After the entire article was matched with keywords, articles with a positive weight were classified positively and those with a negative weight were classified negatively. Finally, the keyword / concept-alone hybrid method applied the keyword scheme first, and if no matches were found, then the classification result of the concept-alone scheme was used.

The results applying the keyword method, concept-alone method, hybrid keyword/concept-alone method, and the index pattern method to information filtering of the 47 articles is shown in table 12. Overall, the human judge identified 25 of the 47 articles to be relevant to the target search patterns. Of these articles, the table indicates the percentage of articles correctly and incorrectly classified by each method. These two percentages do not add up to 100% since the contribution of irrelevant articles is not displayed. Irrelevant articles are defined as articles that should not be retrieved negatively nor positively. The table also decomposes the overall incorrect classification rate into the number of false classifications and the number of missed classifications. False classifications are defined as positive or negative classifications made by the system that are incorrect. In other words, the article was actually classified opposite the prediction or was deemed irrelevant. Missed classifications are defined as articles that the system deemed irrelevant but were classified positively or negatively by the human judge.

Classification Method	Percentage Classified Correctly	Percentage Classified Incorrectly	Number of False Classifications	Number of Missed Classifications
Keyword	52	23	3	8
Concept Alone	40	42	17	3
Keyword / Concept	60	34	13	3
Index Patterns	80	17	5	3

Table 12 : Comparison of Keyword, Concept Alone,
Hybrid, and Index Pattern methods for filtering the "<actor> get <object>" concept.

Table 12 indicates that the index pattern method clearly performed the best, correctly classifying 80% of the articles. The concept alone scheme performed the worst and also had the highest error rate. The error comes primarily from the number of false classifications. The false classification hit rate occurs due to the incorrect disambiguation of word senses; as a result, INFOS incorrectly classifies irrelevant articles as being relevant. The opposite problem appears for the keyword method. In the keyword method, the identification of precise keywords does not suffer from incorrect senses as much as the concept-alone scheme, but does suffer from being too precise. Articles that should be retrieved are missed since keywords, not concepts, are being compared. The hybrid scheme improves upon the concept-alone and keyword methods, but at the cost of higher error carried over from the concept-alone scheme.

The errors made by the index pattern scheme are primarily due to anaphora in the article text. The index pattern scheme does not disambiguation pronouns. Consequently, articles containing sentences like "I am looking for it" will not trigger the index patterns since the word *it* is not disambiguated as falling under one of the index pattern's conceptual categories. In order to disambiguate pronouns and better understand the input text, higher level semantic knowledge such as scripts, plans, and goals are required. The incorporation of this knowledge into INFOS remains an area of future work.

7.4 Chapter Summary

This chapter has described partial parsing via index patterns as a method to increase performance. The highlights of this scheme include:

- Creation of index patterns to map stereotypical uses of language onto their corresponding concepts and classifications. Index patterns are defined as sequences of WordNet concepts.

- Experiments using index patterns for classification resulted in an accuracy rate near 80% while the keyword and conceptual method resulted in an accuracy rate of 60%. The index pattern method clearly outperforms Paice's disambiguation algorithm employed in the case-based reasoning section.

- Although index patterns result in a higher classification rate than Paice's algorithm, the patterns must currently be created by hand and require knowledge of WordNet that may be beyond the casual user. In contrast, Paice's method is automatic and requires no user intervention.

8. Previous and Related Work

A large amount of work has been performed in information retrieval, and just recently attention has been focused on information filtering. To create an intelligent information filter, three major issues must be addressed: (1) A means to model the user's goals, actions, expertise, interests, or behavior, (2) A method to extract key defining features from the input article text or understand the content of the article, and (3) A method to classify the input text based on the defining features from the user model and input text. The following sections describe previous work that involves these issues, and then examines some complete information filtering systems.

8.1 Prior Work - User Modeling

An intelligent news filter must be able to distinguish between articles that are likely to be of interest to the user and those that are likely to be of little interest. This demarcation is highly dependent upon the personal preferences of each user. For example, a user may be interested in all articles pertaining to a particular topic, all articles posted by another user, or any combination of these or other features. While a system could be hard coded to recognize certain features, such a system would lose its utility as user interests change. A more flexible system must be capable of adapting over time to recognize new features.

Several methods to implement adaptive user models are surveyed by Norcio (1991). Early approaches relied upon stereotyping techniques where users are asked questions and each user is placed into an appropriate category (Rich, 1979). However, this technique requires considerable user effort and users may not know the answers that will benefit them most. More powerful and useful techniques focus upon classifying the

expertise and individual differences of each user (Rouse, 1989). However, expertise and differences are not well-defined; consequently, Norcio believes that fuzzy systems are appropriate structures to model users accurately. Although work is still under development, Norcio's goal is to define fuzzy logic that models dynamic systems and human behavior found in the real-world. One approach is to define categories of fuzzy users. Fuzzy user categories may be used for classifying user expertise; for example, in an intelligent tutoring system, a fuzzy student model could adapt the system to the progress of a student.

Neural networks provide another solution to adaptive user modeling (Maren, 1991). Maren first describes the challenges of Human-Computer Interaction. Adaptive models will be necessary for more effective control tasks, tutoring systems, or information retrieval systems. However, very few systems currently allow for adaptive models due to the complexity of the task and the novelty of new technologies such as neural networks. While learning-style methods of user modeling show some promise (Kolb & Fry, 1975), a computer system may be difficult to implement and establish empirical support for these methods. However, Maren argues that the properties of neural networks show the greatest promise in the area of adaptable user models. Self-organizing neural networks (Kohonen, 1989) allow for the automatic reflection of user characteristics and backpropagation networks are capable of generalizing to capture abstract qualities such as attention (Maren 1990). All neural network techniques can be used with incomplete or noisy data.

Additional work with neural network user modeling has been investigated by Chen and Norcio in the UM-net system (Chen & Norcio, 1991) which generates descriptions about software tools. These descriptions are stored in a hierarchical database where each subframe is a specialization of the parent frame. Consequently, the depth of the database determines the degree of detail while the breadth determines the type of information retrieved. Both of these parameters are controlled by a user model neural network.

A technique similar to neural network training is the use of competitive agents (Baclace, 1992). In this approach, agents are sensitive to domain features and become active when these features arise in the input. Each agent will favor some input article, and compete with each other through an economic model. The prediction of an article "costs" each agent, while agents are "paid" depending on how well they predict the user's rating of that article. As the system is used over time, the user's preferred agents will be paid well, while non-preferred agents will lose their capital and be deleted.

8.2 Prior Work - Feature Extraction from Article Text

Before a news article may be intelligently processed, the article must first be understood to some degree by the system. For information filtering, incoming articles must be understood well enough so that the content can be compared with the user model to determine if there is a match. Typically, understanding is demonstrated by the extraction of key features from the text, or by providing a summary of the article. The easiest and most direct method of feature extraction is simply to pull keywords or tokens from the text that match a predefined set of words describing a user's interests (Foltz & Dumais, 1992) or simply to use all of the words in the input article as features (Eberts, 1991; Jennings & Higuchi, 1992). Often, the words are first passed through a stemmer and a stop list. While the keyword approach combined with stemming or stop lists can be effective, it is difficult to predefine all relevant keywords that may occur in a text, or text may be worded in a manner that does not match a keyword. Additionally, since this system has no semantic information, synonyms or similar concepts such as the words "car" and "automobile" are treated as separate entities.

One solution to this problem is a statistical approach, such as Latent Semantic Indexing (LSI), which captures the associations between words and phrases beyond the pairwise independence assumption (Foltz & Dumais, 1992). Latent Semantic Indexing

assumes that some “latent” structure exists in word usage within documents, and that this structure may be statistically approximated. In LSI, a word by document matrix is decomposed into 100 to 300 orthogonal factors that are used to index documents. These documents (and queries) are represented as continuous values along each of the indexing dimensions. This approach allows deeper semantic meaning to be captured than is possible from a surface feature analysis, and even if a document and query have no words in common, the document may be retrieved.

Recently, other statistical approaches have been advocated by natural language researchers such as Charniak (1993). Charniak describes chart parsers and probabilistic grammars to perform the tasks of text understanding and prediction. In a similar vein, Paice has experimented with statistical approaches to generate indices automatically for the back of a book. (Paice, 1989). In this approach, statistically significant noun phrases are extracted for use as book indices. A similar approach is used in INFOS and is explained in further detail in section 5.

More robust approaches to feature extraction have been explored in detail within the field of natural language processing. These approaches incorporate linguistic, connectionist, statistical, or knowledge-based methods of processing (Germain, 1992). Most of the work in this area has concentrated on knowledge-based methods, although statistical and connectionist approaches have recently received more attention. The main advantage of a symbolic, knowledge-based approach is that the input text is understood as a human might understand the text, allowing for many possibilities (Ram, 1992).

One of the earliest knowledge-based approaches to news story understanding is the FRUMP (Fast Reading Understanding and Memory Program) system developed by DeJong (1982). FRUMP is given direct input from UPI news wire stories, processes the story, and provides a summary of the article. Although FRUMP was not designed to perform news filtering, it actually addresses a more difficult problem - that of story

understanding. If stories can be understood, then together with a user model, filtering is made much easier.

The key data structure in FRUMP is the script; scripts capture stereotypical knowledge about the world. Once a script has been recognized, implicit information may then be easily inferred. For example, the political demonstration script predicts that demonstrators arrive, march, police arrive, and there may be arrests. If the input text does not explicitly state all this information, FRUMP is able to infer likely possibilities. Scripts effectively act as top-down predictors, while a substantiator module verifies low-level story details. The appropriate scripts are not recognized by keywords but rather by the meaning of phrases. Once a script has been recognized, the story is processed in terms of conceptual dependency primitive actions and may then be summarized.

While FRUMP has been a successful program capable of dealing with unrestricted textual input, it does have many limitations. A large amount of work must be done defining the lexicon and script knowledge. A more serious problem is that any stories that are not covered by a script cannot be understood. Consequently, truly novel stories or stories that combine many scripts may not be processed.

A more recent work that also performs script based learning to understand and retrieve Usenet news articles is Mauldin's FERRET system (Mauldin, 1991). In Ferret, a query and text articles are parsed into Schank's Conceptual Dependency (CD) theory (Schank, 1977). Intended to be an unambiguous representation for knowledge, once text has been parsed into a CD representation, conceptual comparison is simply a matter of comparing slots and fillers for different values. Statements that are expressed differently in English but are conceptually identical will parse into identical CD structures. In Ferret, input news articles and a search query are all parsed into CD. Then, predefined scripts are compared to the news articles. As in FRUMP, the scripts represent stereotypical sequences of events and information. Articles that match the defined scripts

may then be disambiguated with the script, classified in terms of their content, and matched with the query.

The novel features in Ferret include an online dictionary to augment the understanding process and script learning through genetic algorithms. The online dictionary is used to help determine part-of-speech and other word attributes to supplement the parsing process. The additional knowledge supplied by the dictionary removes some of the brittleness accompanied with knowledge based systems that require large amounts of commonsense knowledge (which must be input by humans). Script learning is accomplished through a genetic algorithm. Parent scripts are selected and mated to produce offspring scripts that are more general (improves recall), more specialized (improves precision), or are combinations of the parents (supports exploration). The offspring scripts that perform well are kept and the process repeated. Performance of FERRET within an astronomy newsgroup exceeded keyword retrieval techniques by 15%.

A more recent knowledge-based approach to textual feature extraction has been implemented in the FANSYS system (Alvarado et. al., 1993; Alvarado & Mock, 1995). FANSYS is designed to understand failure description manuals regarding the Data Management System (DMS) of NASA's Space Station Freedom and perform diagnosis or answer questions; recently FANSYS has also been applied to the failure diagnosis in another domain, the Kuiper Airborne Observatory. In FANSYS, comprehension of input text and questions is performed using the case-based parsing techniques provided by DMAP, a Direct Memory Access Parser (Riesbeck & Martin, 1986; Riesbeck & Schank, 1989). In DMAP, parsing is viewed as a recognition process, i.e., the goal of the parser is to determine which memory structures best organize the input based upon what the parser has already been exposed to. In reading the case describing a gateway failure, the parser will automatically find many of the other case representations it has already seen and use them to help understand the new input. The output of such a parser is the set of memory

structures that have been referenced in the understanding of a new text (such as part of some other case), the new structures added to memory during the parse, and the set of expectations about what will be seen next based on what was just read. This output is then used to retrieve relevant cases for failure diagnosis.

In addition to FANSYS, which uses case-based reasoning for failure diagnosis, case-based reasoning has been shown to be a feasible and useful approach for news story classification and retrieval. Masand showed that increased recall precision was correlated with an increase in the database size (Masand, 1993). In Masand's work, documents were classified using a nearest-neighbor approach for matching terms. A similar approach, but augmented using an AI thesaurus, has been implemented in the CLARIT system for document indexing. In their system, noun phrases are extracted with a parser and used to index text. Their results indicate that sometimes full-text articles can automatically be indexed better than humans. (Evans et. al., 1991). The work in this project incorporates case-based and automatic indexing techniques to filter information.

Another recent knowledge-based approach to text processing has been implemented in the SCISOR system (Jacobs & Rau, 1990). SCISOR is designed to process financial news stories regarding corporate mergers and acquisitions from an on-line news service and extract important information into a structured form. Drawing upon previous approaches and operating upon a much larger scale than previous systems, SCISOR's integrated design includes a topic analyzer filter along with bottom-up and top-down processing. The initial filtering process attempts to classify input stories as definitely relevant, unknown, or definitely irrelevant to a corporate takeover. This is accomplished by keyword filtering for pre specified words such as "buy" or "merger". After the keyword phase, a pattern match is performed to classify the stories further. Unknown stories are then subjected to a more rigorous lexical and conceptual analysis, incorporating bottom-up and top-down processing. The interaction between the bottom-up and top-down components are what distinguish SCISOR from previous systems, as

SCISOR is much more tightly integrated other approaches such as FRUMP. The bottom-up component takes individual words and maps them into a conceptual framework. The top-down component takes the current concept and generates further expectations. Both of these components work in concert as text is parsed. By incorporating both techniques, the bottom-up component can process unique or unexpected input text, while the top-down component can fill in missing gaps of implicit information.

8.3 Prior Work - Document Classification/Filtering Systems[3]

After a user model has been constructed and key features extracted from input articles, an algorithm is necessary to classify the articles. Typically, the classification algorithm will be closely integrated with the feature extraction method, although the two components are modularized in some systems (Jacobs & Rau, 1990). For information filtering, the classifier will simply be determining the interest level of a particular document.

The original news reading program for Unix is RN, short for "Read News". Although simple, RN does contain primitive support for filtering. Upon user request, a "KILL file" can be created that automatically discards messages from particular authors. This feature is useful if a specific individual is flooding a newsgroup with annoying or useless messages. Most other popular newsreaders, such as trn, or tin, also support KILL files while providing additional functionality. Other popular newsreaders such as Netscape's *Netscape* news browser or Fortè's *Agent* have evolved to provide a graphical user interface for greatly improved browsing and selecting, but filtering is limited to watching for threads and KILL files.

---

[3]Links to networked resources about information filtering is maintained by Doug Oard at http://www.ee.umd.edu/medlab/filter/filter.html

One of the predecessors to RN which does contain a more robust filtering component is STRN. STRN allows virtual newsgroups to be created from a selection of Usenet newsgroups, and also ranks articles for filtering. However, the system is strictly keyword driven based on subject and author. Upon reading a message, the user is given the opportunity to rank the article numerically using any scale. Scoring is performed by explicitly adding text to the filter with a score. Manual entry or editing is required, unless the user is willing to allow the entire subject line of an article to enter the filter verbatim. Finally, scoring is additive and linear; for all matching keywords from an article that match words in the filter, the scores corresponding scores are summed. Articles with the highest score are ranked first, the lowest at the bottom. While useful, performance using this scheme is limited since it requires a high amount of user interaction, requires users to score consistently, is undesirable when there are competing keywords that offset each other or when keywords have different meanings in different contexts.

More sophisticated news systems include Riloff and Lehnert's text-skimming approach to classification through the use of Relevancy Signatures (Riloff & Lehnert, 1992). This approach is inspired by the skimming capabilities exhibited by humans in identifying texts relevant to a domain. In their system, input articles from the MUC-3 domain are classified as terrorist or non-terrorist activities. The relevancy signature algorithm first requires training upon a corpus of text to learn a so called relevancy signature. This signature consists of statistics for word/concept pairs that are extracted through a parser. These statistics indicate the frequency of relevant concepts present during training that are then used later for classification. The successful precision of their approach indicates the feasibility of a fast skimming algorithm. Furthermore, the automated approach scales up well to large corpora and may be easily ported to other domains, in contrast to hand-coded knowledge-based approaches. This approach has recently been modified to include a set of retrieved cases to perform classification, achieving even better precision (Riloff, 1993).

In contrast to Riloff and Lehnert's classification schemes, Ram's PIES system processes text at a deeper level to achieve a higher degree of understanding (Ram, 1992), but at the cost of increased complexity. PIES attempts to model the goals and interests of the user to guide the information filtering process. Based upon pre-specified user interests, parts of the parsed knowledge structure can be pruned away in a pre-processing or post-processing phase. The central story processing phase incorporates standard knowledge-based natural language processing techniques (DeJong, 1982).

Another approach to information filtering incorporates the use of rule-based agents to model a user's usage patterns (Stevens, 1992). In Stevens' INFOSCOPE system, pre-defined agents are activated depending on terms from the header or body fields of news messages. As the system is used over time, the agents will model the frequency of textual patterns that appear in the articles that are read. These terms are displayed in an editable dialog box so the user is always aware of what filters are active. These filtering agents may then be easily modified if desired. Additionally, the agents are also capable of learning autonomously based upon user reaction to messages. For example, if a user replies to a message or saves a message, the user is most likely interested in the content of the message and the agent can be updated to reflect this interest. Collectively, these agents allow INFOSCOPE users to create their own virtual newsgroups so that Usenet conforms to their personal structure, rather than forcing them to conform to Usenet structure. Finally, through a GUI, INFOSCOPE improves the task of browsing many messages by organizing messages through threads in a point-and-click interface. Although user-friendly and effective, the main limitation of the filtering component of INFOSCOPE arises from the use of boolean logic and keywords as a central element of processing.

A system similar to INFOSCOPE has been implemented by Sheth (1994) but incorporates a genetic algorithm on top of a keyword based filtering algorithm. In Sheth's NewT system, agents are attuned to various keywords via a weighted score of keywords and suggest whether articles should be read or ignored. Sheth's innovation is to add a

genetic algorithm to control the agents to explore new newsgroups that the user is likely to be interested in. While users like the system, it suffers the same problems as other keyword-based systems; using the right keywords and vocabulary rather than concepts. Additionally, the genetic algorithm was limited to exploring only different newsgroups rather than concepts that agents may have learned.

Another keyword/rule-based news filter has been implemented in the Tapestry system (Goldberg et. al, 1992). Tapestry has its own query language, TQL, somewhat similar to SQL, which is used to specify what types of information should be retrieved or filtered. Consequently, Tapestry has the power of a full database retrieval system, but it is also limited by the vocabulary and keywords that are encoded into the boolean queries. The major innovation in Tapestry is the support of collaborative filtering. Collaborative filtering (often called social filtering) refers to the collaboration of many users to aid in the filtering process. This is accomplished by having users annotate articles as they are read. This information is made public, and becomes an input for the text filter of other users. As a result, other users may decide to read an article based upon the reaction of their peers; e.g., user A may choose to read articles only examined by user B or user C.

Championed by the MIT multimedia laboratory, collaborative filtering has recently become a popular area of research. Collaborative systems for filtering mail, Usenet news, and WWW documents are currently under investigation (Brewer & Johnson, 1994; Lashkari et. al., 1994). However, instead of requiring explicit direction by a user to determine how collaborative filtering is performed, keyword features are automatically extracted from input articles and used to filter incoming data transparently. Results from these systems indicate that collaborative methods have all improved filtering performance.

Other popular keyword based systems for News filtering include Wide Area Information Servers (Kahle, 1991), and World Wide Web servers (Berners-Lee et. al., 1992). These systems contain archives of newsgroups and an interface to perform

boolean search queries. One popular system is the DejaNews[4] news service (Crowe, 1995) that archives over 4 gigabytes of news data and allows users to search for articles based upon keywords in the subject or authors headings. A sample search could be performed to look for all articles from a comp newsgroup that contain the words "information" and "filtering". Note that these are passive search systems rather than active agent-based systems; the user must explicitly enter what material she wishes to search for. Active filtering systems learn user interests based upon user feedback and suggest new messages for the user to browse.

One keyword system that also requires direct user instruction to perform filtering but also provides personalized filtering, is SIFT, the Stanford Information Filtering Tool (Yan & Garcia-Molina, 1994). SIFT is an email service; users email their profiles and filtering requests to the server in a boolean format, and SIFT will efficiently search through new articles that match these requests using an inverted index of profiles. An inverted index is simply a reverse index where the indices are composed of individual words, and the data is a list of articles that contains that index word. In this manner, all articles containing a specific word can quickly be found. When keywords within news articles are found that satisfy the query, these matching articles are e-mailed to the user. In this fashion, the user can passively wait for incoming articles of interest, rather than actively checking on his or her own. While very convenient, SIFT is limited to matching keywords and does require users to formulate their own queries.

In addition to keyword and knowledge-based approaches to information filtering, neural networks may also be used to classify incoming articles. Eberts examines the direct approach of feeding raw words into a feedforward backpropagation network (Eberts, 1991). The output units indicate the category of the article. This approach is limited to a shallow understanding of the text by relating together all of the words present

---

[4]The DejaNews news service may be reached at http://www.dejanews.com.

in the article. Furthermore, since all knowledge is captured in neuron weights, it is extremely difficult for a user to modify the model created of that user directly. Experiments were conducted using the header alone and the entire body of the text. In their tests, the header alone was sufficient to classify articles correctly in most instances.

A larger scale neural network news filtering system has been implemented by Jennings and Higuchi (Jennings & Higuchi, 1991). Their approach uses a user model semantic network to identify key words and concepts. In their system, articles are read and marked as rejected or accepted. At the completion of a session, these articles are then examined by extracting the common features and a semantic network is created. This network is then compared to the user model belief network, and nodes are added or deleted as appropriate. The feature extraction process simply uses the first 300 words of the article as features, eliminating commonly used words. Under the assumption that the header and initial paragraph are representative of the article's contents, and that most articles are fairly short, this approach is feasible. However, due to the structure of a news article, additional emphasis is placed on "Subject" or "From" header lines. At the end of the extraction phase, a list of up to 300 words are collected from each article. These words are then connected to each other in a network whose connection strengths indicates the frequency of the appearance of the words. Consequently, words that are frequently associated in articles read by the user will be strongly connected. The resulting network comprises the user model neural network. To classify a new article, its features are extracted in a similar manner and matching nodes in the user model network are activated. Adjacent nodes are then "fired" if their input energy is greater than a threshold value. Here, the energy is computed by summing the connection weight by the value of the neighboring nodes. The process is continued for several iterations, at which point the set of active nodes is summed to determine the ranking of interest of the article. Initial experiments with the system have indicated promise with their approach, and the

construction of a well-designed user interface that also supports keyword search further increases the utility of the system.

In a similar vein, Lang has also experimented with neural network-based filters through a World-Wide Web based newsreader named NewsWeeder[5] (Lang, 1994). However, instead of directly predicting the category of an article, Lang used a neural network along with Singular Value Decomposition (SVD - a straightforward linear transformation) to reduce the original search space of the raw text into a new, discrimination-dense representation. In Lang's experiments, neural networks were capable of reducing the search space and finding relevant terms to predict which newsgroup articles originated from with 75% accuracy.

While successful, Lang's recent work with NewsWeeder (Lang, 1995) focuses on news filtering using the Minimum Description Length (MDL) principle. MDL provides a probabilistic model for computing the most likely hypothesis based upon Bayes' Rule. NewsWeeder assumes conditional independence among the terms or keywords parsed from articles and uses these features to compute the most likely rating class, based on the probability mixture of each term for a specific category and each term in the global distribution. In operation, users read the messages and rate each one from 1-5, 1 being of interest and 5 being of disinterest. Articles were then decomposed into tokens, or keywords. These tokens include words, punctuation, authors, or newsgroup names. The user feedback determined the probabilities of the classes, and the tokens supply the evidence upon which MDL can be performed to classify future articles. Lang found that NewsWeeder performed well, with a precision (relevant documents retrieved compared to irrelevant documents) up to 65%, up to 20% better than the popular term-frequency/inverse-document-frequency approach.

---

[5] The NewsWeeder home page is http://anther.learning.cs.cmu.edu/ifhome.html

8.4 Chapter Summary

This chapter has provided an overview of other systems which perform user modeling, feature extraction, or information filtering. The highlights of these systems include:

- Use of neural networks, competitive agents, and fuzzy categorization to model users.

- Feature extraction through latent semantic indexing and statistical techniques in addition to knowledge-based techniques employing scripts and case-based reasoning.

- A description of other information filtering systems that are based upon keywords, collaborative data, genetic algorithms, neural networks, the WWW, and probabilistic models.

9. Current Status and Future Work

The prototype of INFOS has been implemented on a Sun Sparc 10 workstation running SunOS 4.1.3 and is comprised of approximately 150K of C source code. Approximately 10 megabytes of free disk space are required for INFOS storage files, and approximately 20 megabytes of disk space are required to store WordNet. At least 16 megabytes of RAM are required to run the system.

In addition to the SunOS version, early implementations of INFOS have also been ported to DECstations. Since WordNet is available on Unix, Macintosh, and PC machines, a porting of the code is possible to all major platforms. Moreover, since INFOS produces only textual output, the system can be run via modem or dumb terminal.

To download the system and view the data files used for the experiments reported in this thesis, direct your World Wide Web browser to::

http://phobos.cs.ucdavis.edu:8001/~mock/INFOS/infos.html

In terms of connectivity, INFOS runs as a news client and connects to any standard NNTP News server, allowing access to news articles from virtually any machine. Future versions of INFOS may require the system to run as a server to reduce user wait time or to operate in different domains. The applications of INFOS to the domain of WWW filtering and improvements to INFOS which must be examined in the future are described in this section.

9.1 Application of Filtering Algorithms to the WWW

In addition to filtering news documents, the algorithms implemented in INFOS are also applicable to any domain with a stream of incoming data; in particular, textual data. An initial study has been performed to apply the filtering techniques to World-Wide Web (WWW) browsing. In this scenario, the goal is to identify web sites likely to be of interest based upon web sites the user has visited. Work in this area is currently under investigation and the results reported here are only preliminary. However, the preliminary results have been promising and indicate that the mechanisms employed in INFOS are general enough to apply to other domains.

9.1.1 Background Information on the WWW

The size of the World Wide Web is constantly growing at an astounding rate. The Lycos search service[6] estimates that the number of WWW pages has grown from 5 million to 6.89 million pages during the months of April to June of 1995, and will reach 10 million pages by 1996. Although search engines are currently the most popular tools for navigating webspace, the large volume of pages in existence also presents an excellent area where information filtering is applicable. By building up a profile of user interests, off-line searches can find new pages likely to be of interest and bring them to the attention of the user. In contrast to finding information of interest, filtering systems for the WWW may also be used exclusively to reject information. For example, a rejection-based filter may be used to censor the WWW to prevent children from accessing adult information. Recent interest in rejection-based filters has grown due to proposed legislation (such as the Exon bill) that seeks to regulate the internet.

---

[6]http://www.lycos.com

One of the first projects in filtering WWW documents is the WebHunter[7] system developed at the MIT Media Lab. WebHunter is based upon collaborative filtering of entire URL's. Based upon a user's own pages of interest (which can be determined from a user's hot-list), other pages are recommended that other users with similar interests have been interested in. To date, user feedback for WebHunter has been positive and the system continues to run, although results on precision and accuracy have not yet been available.

One of the benefits of collaborative filtering is that semantic content, graphics, audio, and animated media are all factors that contribute to the filtering process since people consider all of these features in making recommendations. Additionally, WWW documents can also be filtered via traditional keyword approaches by extracting textual keywords from the web documents. The keyword approach is the method implemented in the WebWatcher system (Armstrong et. al., 1995). In WebWatcher, users input words defining the goals they are looking for. Keywords are extracted from the web documents linked off the current page, and then these pages are classified with respect to the goal. When searching for this goal, links that the system believes will lead to the goal are highlighted, aiding the user in navigating webspace. The learning methods employed to compare web documents with the user's goals included a linear weighting of features, statistics of individual words, and the tf-idf technique. All techniques worked better than random, and the linear weighting method predicted the user-selected link in its top three choices in 54% of experimental cases.

---

[7]http://www.webhound.www.media.mit.edu/projects/webhound/doc/Webhound.html

9.1.2 WWW Filtering with INFOS

To examine the potential of INFOS to filter WWW documents, 70 web pages were selected at random from the Yahoo[8] catalog. Seven web-savvy volunteers examined and classified each page as being of interest, ambivalent, or disinterest. INFOS was trained upon 35 of the web pages using the Global Hill Climbing and CBR techniques, and then tested upon the remaining 35 web pages. The results are shown in table 13. Since there are no authors or subject headings for WWW documents, only collaborative data and the features extracted from the body of each web page were used for filtering.

Classification Method	**Percentage Classified Correctly**	**Percentage Classified Unknown**	**Percentage Classified Incorrectly**
Global Hill Climbing: Textbody Alone	35.4	45.9	18.7
Global Hill Climbing: Collaborative Alone	32.7	60.3	7.0
Global Hill Climbing: Combined	41.2	42.7	16.0
Case-Based Reasoning Alone	37.0	49.7	13.3
Combined CBR and Global Hill Climbing	36.9	45.4	18.2

Table 13 : Filtering results of WWW Documents

The collaborative method resulted in the lowest error rate at 7.0 and a correct classification rate of 33%. Since the collaborative method had the lowest error rate, the combined Global Hill Climbing method weighted collaborative data 70% and textbody features 30% to favor the collaborative method. Nevertheless, features from the textbody introduced error in both the hill climbing and CBR schemes. The error may likely come from the lack of data. Training upon 35 web documents selected at random from millions

[8]http://www.yahoo.com

does not provide an accurate sample of other web pages. Consequently, users may need to directly edit their models or examine many more documents in order to search the web space accurately. Nevertheless, using just 35 pages, all methods do classify better than random (32% correct but 30% error).

The collaborative method alone appears the most promising for this domain. Since it is difficult for a single user to examine many web pages, the collaborative scheme effectively splits the work by having many users examine portions of the web space for each other. Furthermore, the proliferation of audio and graphical features on the WWW are not indexed using the keyword schemes. Consequently, an important aspect of each web page is lost in the keyword model. Future work in indexing non-textual data may significantly improve filtering performance.

9.2 Future Work and Implementation Considerations

In addition to the application of INFOS to WWW filtering, a large amount of work still needs to be done upon the basic algorithms of INFOS and the application of INFOS to other domains. Rather than focus on speed and efficiency, the existing version of INFOS has been designed as a prototype to test the ideas presented in this thesis. For example, linked lists and array data structures are used for their simplicity, while a more complex hash table implementation will reduce search to constant time rather than linear or logarithmic. Similarly, memory-consuming arrays are used in several functions in lieu of dynamic data structures that use only the space required. In addition to these implementation issues, user interface and resource issues also need to be addressed to make INFOS usable in practice. Moreover, conceptual processes such as planning and reasoning may be implemented to better understand and classify input articles. Finally, in addition to being applicable to Usenet news and the WWW, the algorithms and ideas behind INFOS may also be applicable to Intelligent Tutoring Systems.

9.2.1 Time Required for Filtering

A major issue with the existing system is that processing messages into the user model and performing the filtering process is relatively slow. The bottleneck is the WordNet lookup phase. While the global hill climbing scheme and memory retrieval is fairly quick, each sentence takes between 1 to 10 seconds to process in WordNet so that the word sense hierarchies can be retrieved. As a result, each message requires anywhere between 20 to 300 seconds to process when filtering or updating the user model. This time wait is clearly unacceptable for users to endure.

Although a more efficient implementation will speed up processing, the optimal solution to this problem is to perform the filter and update processes off-line. This is simple to perform for the user model update since INFOS already stores all of the read messages in a file. Currently, INFOS updates the user model after the user quits reading the selected newsgroup, but the update could be delayed until after the user is finished reading all messages. Additionally, the update can be performed as a background process.

To perform the filtering process off-line is slightly more complicated since this requires INFOS to run as a server, constantly running in the background. Ideally, at periods of low CPU usage (such as late at night or early morning) the system will filter new news articles for each user. The filtered articles will then be ready in the morning for users to peruse. While this organization will dramatically reduce user wait time, users will have the drawback of only being able to read filtered articles after an update has been performed, or be forced to wait for the update to finish. As a compromise, if users wish classification to proceed quickly, INFOS is currently capable of performing only the global hill climbing filtering scheme.

9.2.2 User Interface

At a high level, the entire INFOS system can be considered one large user interface. The filtering system adapts the data into an organization easily managed by the user, rather than forcing the user adapting himself to the way raw data is organized. The actual interface in INFOS is text-based rather than graphical. The rationale behind this approach is the faster development time and increased portability across a variety of platforms. Moreover, a text-based interface allows easier access via modems without the need for SLIP or PPP.

Nevertheless, graphical user interfaces do allow data to be displayed, browsed, and edited more efficiently. In particular, color, font styles, graphical depiction of links, and control over text layout adds additional power to the browsing process. The use of graphics also allows for new ways of representing data; for example, articles can be represented by icons or spheres, where the size of the sphere indicates the size of the article, and the color could indicate content or filtering suggestions. To create an even more flexible and usable system, the filtering techniques employed in INFOS should be combined with a graphical user interface.

9.2.3 Client-Server vs. Peer-to-Peer Communications

The current architecture of INFOS is designed under a client-server architecture. Each user may run INFOS on a different machine, but if they wish to share collaborative data, all users must share the same file system. A more ideal system would run in a peer-to-peer fashion so that users can run on separate file system and separate machines. Each user could then run on distinct machines and specify other users that they wish to communicate with in order to exchange collaborative data. This will require each user to run a daemon to handle communications, but this could be performed easily through a

WWW server. In the event that other servers are down or inaccessible, caching techniques can supply temporary data. As processing power becomes more readily available to individual users, peer-to-peer communications will become more popular since network traffic bottlenecks are avoided and CPU power is more efficiently utilized.

9.2.4 Self-Modifying Parameters

Throughout the experiments conducted in this work, the weighting of parameters was determined by determining experimentally how well individual methods performed (e.g., author alone, text alone, collaborative alone, etc.) and then combining the features so that the most predictive and accurate ones carried the most weight. This process could be automated if INFOS tested upon previously read articles using the individual methods and then updated the weights to reflect the most accurate method. In effect, hill climbing is being performed on the parameters themselves. As a result, INFOS users would be guaranteed that the current set of weights accurately reflects the features that are most predictive for them. The cost of the self-modifying approach is an increase in the amount of computation that must be performed, but the computation could be performed off-line.

9.2.5 Scripts, Plans, and Goals to Improve Understanding

As described in chapter 7 regarding knowledge understanding, the incorporation of scripts, plans, and goals (Schank, 1977) is the next step towards a more complete understanding of input articles. The use of WordNet in INFOS is the first step towards this knowledge since WordNet facilitates bottom-up indexing. Based upon the meaning of a word, these knowledge structures can be indexed. However, the actual structures and the top-down mechanisms that govern how the structures fit together are not yet implemented in INFOS.

Scripts have already been demonstrated in Ferret (Mauldin, 1991) to be effective in understanding news stories. In addition to scripts, plans need to be represented in order to understand novel articles that do not fall within the domain of scripts. Given a goal, planning attempts to link together chains of reasoning so that the goal can be achieved. This type of planning knowledge is necessary to understand how and why an author is piecing together information in an article. In addition to understanding how articles are fit together, plans and goals can also be used in the user model. If INFOS can understand a user's goal (e.g., finding all articles relating to the Bosnian-Serb conflict), plans can be executed to meet that goal (search appropriate newsgroups or WWW sites).

To reason with scripts, plans, and goals requires a large amount of commonsense knowledge. Knowledge is required to parse texts, create plans, and make inferences. Traditionally, the commonsense knowledge problem has been the bottleneck that prevents knowledge based systems from becoming practical applications. Manual input has been the only efficient method of entering commonsense knowledge, and it is a monumental task to enter enough knowledge for a system to be useful. INFOS sidesteps this problem by performing partial understanding and by linking understanding with keyword systems.

One possible solution to the knowledge problem is the CYC knowledge base (Lenat, 1995). CYC is a large knowledge base containing approximately 10^6 commonsense axioms that have been painstakingly built by hand over the past ten years. With access to this large database of knowledge, it may be possible to generate accurate goals, plans, and inferences. General parsers can be implemented, models of domains created, and general textual articles understood. Although CYC is still under construction and its availability is limited to commercial groups, CYC applications are now in the prototype stage. Currently, CYC appears to be the AI community's best hope for a general semantic backbone that can be used for information filtering and a large variety of other applications.

9.2.6 Subjective Comprehension of Input Articles

There are many different types of Usenet news articles. Information dissemination is one popular type of article. Articles in this class deal with postings about current events or notifications regarding a new event or service. Other types of articles include calls for help and advice. These types of narrative articles can be understood via scripts, goals and plans. However, one of the most prevalent types of Usenet articles that is difficult to understand through scripts, plans, or goals is the *commentary*. When a user is posting an article, the user is typically giving his or her opinion about the topic at hand. To understand editorial articles fully requires knowledge about what the author is discussing in addition to knowledge about how editorial arguments are constructed (Alvarado, 1990).

Comprehension of input text for news filtering must also be influenced by the *ideological perspective* (Carbonell, 1981) that the system may have about a given domain. That is, the system must attempt to understand news articles and relate their conceptual content to the user's beliefs and justifications involving related and/or similar articles. If INFOS reads descriptions that are inconsistent with the model it has constructed for the user's beliefs, then INFOS must be able to recognize and use the inconsistency as a feature to classifying articles. Depending upon user preferences, a user may be interested in articles that support her beliefs or also articles that attack her beliefs. In order to account for this process of subjective comprehension, INFOS must develop an ideology for the user as well as strategies for determining inconsistencies between beliefs in long-term memory and input text. Processes for tracking counterarguments will also be necessary to understand how message threads of editorial replies are constructed. These counterargument strategies should be based on the argument-planning knowledge underlying the taxonomy of argument units proposed by Alvarado (1990).

9.2.7 Filtering and Intelligent Tutoring Systems

In the domain of Intelligent Tutoring Systems (ITS), a crucial component that must be addressed is a model of the student's knowledge and behavior. This model must be flexible enough to accommodate changing interest or behavior of the students. The problem of modeling a student's changing interests and newly acquired knowledge is similar to the problem of adapting to changing user interests for news reading. Consequently, the same algorithms and modeling techniques applied to filtering systems may also be applied to an ITS. The modeling techniques currently under investigation include learning through genetic algorithms and feedback from the student. The GA and feedback is then used to guide future tutoring interactions.

Under construction as the prototype system "Shadow," the system is based upon INFOS (Mock & Vemuri, 1994) and the educational system GAITS (Quafafou, 1994). In GAITS, supervised teaching techniques are employed where the teacher assigns a pedagogical objective to each student before teaching begins. In order to achieve this objective, the tutor interacts with the learner using predefined dialogues. Dialogues are composed of a *Header* and a *Body*. The Header contains *prerequisite knowledge* that must be mastered by the student to consider this dialogue as a candidate, *taught knowledge* that comprises the actual lesson, a *learning level*, and a *learning strategy* to present the lesson. The Body represents the teaching materials that define the type of the interaction (e.g., exposition, question/answering, game, etc.).

Shadow's interaction is based upon tutoring cycles that start by filtering a set of dialogues from the Dialogues Data Base. A dialogue passes the filter if its prerequisite knowledge is satisfied. Next, Shadow predicts which candidate dialogue is most likely to be of interest, taking into account the student level, the current sub-goal, and the student's prior positive or negative feedback. At the end of a tutoring session with this dialogue,

the student gives feedback whether or not the dialogue was effective (accepted or rejected) and the student model is updated by incrementing accepted or rejected counters corresponding to the dialogue's knowledge. The feedback process is identical to the global hill climbing scheme implemented in INFOS. If necessary, new sub-goals are defined and new dialogues filtered. The end of a tutoring cycle corresponds to the achievement of the pedagogical goal and the evolution of individuals used for prediction.

The student model contains a population of individuals, where an individual is a table containing taught knowledge, a learning level, a learning strategy, and the number of times the student has accepted and rejected this knowledge. Given a population of tables of these tuples, Shadow employs a genetic algorithm upon the population to maximize individuals whose features received positive feedback. The genetic algorithm is employed since it has been shown to be effective in exploring and finding global optima in complex search spaces.

In addition to the use of global hill climbing and genetic algorithms, collaborative learning may also help to select appropriate teaching strategies when groups of students are learning a goal. Given a group of students and a set of lessons the students must learn, not all students will learn best with the same strategy. Some students may learn best when actively involved, when the material is presented as a game, when simply taught the subject in a standard lecture presentation, etc. By examining collaborative teaching data, students can be grouped together according to those strategies that are most effective for them to learn a lesson. For example, if students X and Y both learned a lesson well with teaching strategy S_1, and then student X learns a new lesson well with strategy S_2, then strategy S_2 may also be an effective method to teach the lesson to student Y.

The ideas presented above for Shadow are experimental. A prototype is currently under construction to validate the methodology. However we are optimistic of Shadow's performance based upon promising results from both INFOS and GAITS.

9.3 Chapter Summary

This chapter has provided an overview the current status of INFOS, applications of INFOS to the WWW, and future work which will increase the power and usability of the system. The highlights of this work and improvements include:

- INFOS may be applicable to filter WWW documents. In an experiment with randomly selected web pages, INFOS was capable of correctly classifying pages of interest with 33% accuracy and 7% error using collaborative features. The applicability of INFOS to other domains indicates the generality of the algorithms implemented in INFOS.

- Work is ongoing to incorporate the ideas in INFOS to selecting appropriate dialogues for Intelligent Tutoring Systems.

- Future work is required to incorporate a graphical user interface, support client-to-client communications, automatically modify weight parameters in the global hill climbing model, incorporate scripts, plans, and goals to improve understanding, and provide a model for the subjective comprehension of news articles.

10. Conclusion

As the information age grows in scale, the amount of incoming data becomes too large for humans to handle. The internet has been growing a tremendous rate. Gigabytes of news articles flow through the internet daily, and World Wide Web pages number around 10 million. The central issue in this thesis addresses methods to model user interests automatically so that this data, Usenet news articles in particular, can be filtered intelligently. However, in order to be a useful tool, the user model must be capable of adapting to user interests, articles must be displayed to give as much information as possible so users can intelligently browse and select articles to read, users must be capable of modifying and understanding the user model constructed for them, and the news filtering system must give accurate predictions.

Previous work to address the information filtering problem has examined either keyword or knowledge-based approaches. Knowledge-based systems have the advantage of analyzing input text in detail, but at the cost of computational complexity and the difficulty of scaling up to many domains or domains of large scale. In contrast, statistical and keyword approaches scale up readily but are limited to a shallower understanding of the input. A hybrid system that integrates all of these approaches improves accuracy and provides scalability along with domain knowledge. To validate this claim, the hybrid approach has been implemented in the INFOS system.

The major contribution of this thesis lies in the further exploration of information filtering, currently a new field that has yet to be fully examined. In particular, this thesis addresses issues in user interfaces, information content, information retrieval, and hybrid methods for information filtering. The contributions are made in the following areas:

- This thesis has investigated user behavior with respect to browsing and reading messages. This work shows that a large volume of articles causes readers to miss many messages that they are interested in reading. In addition to information overload, current browsers also present data in a poor manner, displaying only author and subject data, but not data regarding the actual content of the message. This evidence supports the need for an information filter and improved methods for browsing news articles. Along with information filtering, INFOS displays the first line of each text article and selected keywords to display some information about each article's content.

- A hybrid filtering scheme composed of hill climbing, case-based reasoning, and genetic algorithms is proposed to address both the information filtering problem and user-interface issues. Experimental results show that the hybrid scheme performs better than any of the individual methods alone, validating the use of hybrid techniques to increase performance.

- A keyword approach named Global Hill Climbing performed well for information filtering, resulting in a correct classification rate of approximately 50% and an error rate of 7%. A case-based approach based on WordNet performed slightly worse with a correct classification rate of approximately 40% and an error rate of 10%. However, the hybrid scheme combining both Global Hill Climbing and case-based reasoning outperformed either method used alone, with approximately 60% of the articles correctly classified. However, the disambiguation problem associated with the case-based approach did increase the error rate of the hybrid scheme slightly to 12%.

- Hybrid schemes for information filtering that combine genetic algorithms with the global hill climbing scheme also outperform the global hill climbing method alone by

up to 12%. The genetic algorithm helps INFOS create a user model that explores other areas of the search space.

- The hybrid scheme strikes a middle ground between keyword based techniques and knowledge-based information retrieval techniques. This scheme allows INFOS to scale up to large domains using keyword techniques while still retaining conceptual indices and some degree of commonsense knowledge. Scalability has been demonstrated by applying INFOS to several newsgroups without pre-defining specific knowledge.

- The incorporation of index patterns to recognize concepts at the phrase and sentence level further increases performance over keyword and word-level filtering methods. Experiments indicate that index patterns may result in as much as a 20% improvement in accuracy. However, unlike the keyword and word-level methods, index patterns must be created carefully by the user and requires knowledge about the architecture of WordNet's knowledge base.

- The case-based approach used for the information filtering engine is also applicable to document retrieval. Through a hybrid approach, documents for the Time database were retrieved with a 12% higher recall rate than the tf-idf method.

- The filtering algorithm is applicable across a wide range of domains involving large streams of textual data, including news filtering, tutoring systems, or World Wide Web filtering. Collaborative filtering is particularly useful for WWW documents because people incorporate visual and auditory features in their reviews while the text-based algorithms do not.

- Experimental results confirmed that collaborative filtering is a useful tool for improving performance. In particular, datasets regarding new or unknown information were filtered best using collaborative features.

At the core, the architecture of INFOS is based upon the global hill climbing keyword method using features extracted from the article subject, author, textbody, and collaborative results. Alone, this method results in a model easily modifiable by the user that performs at a low error rate. The keyword method is expanded by incorporating automatic indexing techniques coupled with the general knowledge-base contained in WordNet to extract noun and verb phrase indices of articles. These conceptual indices then index the articles in a case-based memory. The case-based memory groups the cases hierarchically in memory, to facilitate quick case retrieval. Additionally, the hierarchy provides a framework for generalization and inferencing. Prior cases act as guides for categorizing new articles when the keyword method fails. Finally, a genetic algorithm component is applied across user models to explore other areas that may be of interest to the user.

Through the use of hybrid techniques, this thesis presents and validates a novel and more powerful approach to information filtering than has been previously explored. With the exception of the index pattern method, all filters described in this work adapt to user interests without the need for explicit user programming. However, as other methods for representing commonsense knowledge become available, it may be possible for index patterns and other conceptual structures to be created automatically. Tools with this amount of depth, power, and flexibility will be necessary to meet the rapidly changing and demanding needs of users as the information age continues to grow.

11. References

Alshawi, H. (1987). Processing Dictionary Definitions with Phrasal Pattern Hierarchies. *Computational Linguistics*, **13** (3), pp. 195-202.

Alvarado, S. J. (1990). Understanding Editorial Text: A Computer Model of Argument Comprehension. Boston, MA: Kluwer Academic Publishers.

Alvarado, S., Braun, R., Mock, K. (1993). *FANSYS: A Computer Model of Text Comprehension and Question Answering for Failure Analysis* (Tech Rep. No. CSE-93-4). University of California, Davis, Department of Computer Science.

Alvarado, S., & Mock, K. (1995). Comprehension and Retrieval of Failure Cases in Airborne Observatories." *Proceedings of the 1995 Goddard Conference of Artificial Intelligence and Emerging Information Technologies*. NASA Conference Publication 3296.

Armstrong, R., Freitag, D., Joachims, T. & Mitchell, T. (1995). WebWatcher: A Learning Apprentice for the World Wide Web. AAAI Spring Symposium on Information Gathering from Hetereogenous, Distributed Environments. Menlo Park, CA: AAAI Press.

Baclace, P.E. (1992). Competitive Agents for Information Filtering. *Communications of the ACM*, **35** (12), pp. 50.

Berners-Lee, T. Cailliau, R. Groff, J.F. & Pollerman, B. (1992). World-Wide Web: The information universe. *Electronic Networking: Research, Applications, and Policy,* **1** (2), pp. 52-58.

Brewer, R.S. & Johnson, P.M. (1994). *Toward Collaborative Knowledge Management within Large, Dynamically Structured Information Systems.* University of Hawaii, Department of Information and Computer Sciences, Collaborative Software Development Laboratory. (Available from WWW: http://www.ics.hawaii.edu/~csdl/urn)

Callan, J.P. Croft, W.B. (1993). An Approach to Incorporating CBR Concepts in IR Systems. *Proceedings of the 1993 Spring Symposium on Case-Based Reasoning and Information Retrieval*, AAAI Press, pp 28-32.

Carbonell, J. (1981). Overview of Politics. In R.C. Schank and C.K. Riesbeck, Inside Computer Understanding, Hillsdale, NJ: Lawrence Erlbaum, pp. 259-307.

Charniak, E. (1993). Statistical Language Learning. Cambridge, MA:MIT Press.

Chen, Q. & Norcio, A.F. (1991). A Neural Network Approach for User Modeling. *Proceedings of the 1991 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 1429-1434.

Collis, K.F. (1980). Levels of Cognitive Functioning and Selected Curriculum Areas. In J. Kirby and J. Biggs (Eds.), Cognition, Develoment, and Instruction. New York, NY: Academic Press, pp. 65-89.

Crowe, E. P. (1995, October 3). Usenet Sleuth: DejaNews. *Computer Currents*, **13** (10), pp. 88-89.

DeJong, G. (1982). An Overview of the FRUMP System. In W.G. Lehnert & M. H. Ringle (Eds.), Strategies for Natural Language Processing, Hillsdale, NJ: Lawrence Erlbaum, pp. 149-174.

Eberts, R. (1991). Knowledge Acquisition Using Neural Networks for Intelligent Interface Design. *Proceedings of the 1991 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 1331-1335.

Edmundsun, H.P. (1969). New methods in automatic extracting. *Journal of the ACM*, **16** (2), pp. 265-285.

Evans, D.A. Ginther-Webster, K. Hart, M. Lefferts, R.G. Monarch, I.A. (1991). Automatic Indexing Using Selective NLP and First-Order Thesauri. *Proceedings of the Intelligent Text and Image Handling Conference*, Barcelona, Spain. pp 624-643.

Foltz, P.W. & Dumais, S.T. (1992). Personalized Information Delivery: An Analysis of Information Filtering Methods. *Communications of the ACM*, **35** (12), pp. 51-60.

Furnas, G.W. Landauer, T.K. Gomez, L.M. Dumais, S.T. (1987). The Vocabulary Problem in Human-System Communications, *Communications of the ACM*, **30** (11), pp. 964-971.

Germain, E. (1992). Introducing Natural Language Processing. *AI Expert*, **7** (8), pp. 30-35.

Goldberg, D.E. (1989). Genetic algorithms in search, optimization, and machine learning. Reading, MA: Addison-Wesley Publishing Company.

Goldberg, D., Nichols, D., Oki, B., Terry, D. (1992). Using Collaborative Filtering to Weave an Information Tapestry. *Communications of the ACM*, **35** (12), pp. 61-70.

Hearst, M.A. (1993). Cases as Structured Indexes for Full-Length Documents. *Proceedings of the 1993 Spring Symposium on Case-Based Reasoning and Information Retrieval*, AAAI Press, pp. 140-144.

Holland, J. (1975). Adaptation in natural and artificial systems. Ann Arbor, MI: University of Michigan Press.

Jacobs, P.S. & Rau, L.F. (1990). SCISOR: Extracting Information from On-line News. *Communications of the ACM*, **33** (11), pp. 88-97.

Jennings, A. & Higuchi, H. (1992). A Personal News Service Based on a User Model Neural Network. *IEICE Transactions Inf. & Systems*, **E75** D(2), pp. 198-209.

Kahle, B. (1991). An Information System for Corporate Users: Wide Area Information Servers (Tech. Rep. No. TMC-99). Thinking Machines Corporation.

Klapp, O. E. (1986). Overload and Boredom, New York, NY: Greenwood Press.

Kohonen, T. (1989). Self-Organization and Associative Memory, New York, NY: Springer-Verlag.

Kolb, D.A., & Fry, R. (1975). Toward an applied theory of experiential learning. In C.L. Cooper (Ed.), Theories of Group Processes, New York, NY: Wiley, pp. 33-57.

Kolodner, J. (1983). Maintaining Organization in a Dynamic Long-Term Memory. *Cognitive Science*, **7**, pp. 243-280.

Kolodner, J. (1983). Reconstructive Memory: A Computer Model. *Cognitive Science*, **7**, pp. 281-328.

Kolodner, J. (1988). Retrieving Events from a Case Memory: A Parallel Implementation. *Proceedings on a Workshop on Case-Based Reasoning*, pp. 233-249.

Kolodner, J. (1993). Case-Based Reasoning. San Mateo, CA: Morgan Kaufmann Publishers.

Lang, K. (1994). *NewsWeeder: An Adaptive Multi-User Text Filter*. Carnegie Mellon University, Department of Computer Science. (Available from WWW: http://anther.learning.cs.cmu.edu/res_sum.ps)

Lang, K. (1995). NewsWeeder: Learning to Filter Netnews. *Proceedings of the Twelfth International Machine Learning Conference.*

Lashkari, Y., Metral, M., & Maes, P. (1994). Collaborative Interface Agents. *Proceedings of the Twelfth National Conference on Artificial Intelligence,* pp. 444-449.

Lehnert, W. & Sundheim, B. (1991). A Performance Evaluation of Text-Analysis Technologies. *AI Magazine*, **12** (3), pp. 81-94.

Lenat, D.B. (1995). CYC: A Large-Scale Investment in Knowledge Infrastructure. *Communications of the ACM*, **38** (11), pp. 32-38.

Maren, A. J., Harston, C.T., & Pap, R.M. (1990). Handbook of Neural Computing Applications, New York, NY: Academic Press.

Maren, A. J. (1991). Neural Networks for Enhanced Human-Computer Interactions. *Proceedings of the Second Workshop on Neural Networks: Academic/Industrial/NASA/Defense*, Auburn: AL.

Masand, B. (1993). Effects of Query and Database Sizes on Classification of News Stories using Memory Based Reasoning. *Proceedings of the 1993 Spring Symposium on Case-Based Reasoning and Information Retrieval*, AAAI Press, pp. 63-77.

Mauldin, M. L. (1991). Conceptual Information Retrieval: A case study in Adaptive Partial Parsing. Kluwer Academic Publishers. Norwell, MA.

Miller, G., Chodorow, M., Landers, S., Leacock, C. & Thomas, R. (1994). Using a semantic concordance for sense identification. *ARPA Workshop on Human Language Technology*. Plainsboro, NJ.

Miller, G.A. (1995). WordNet: A Lexical Database for English. *Communications of the ACM*, **38** (11), pp. 39-41.

Mock, K., Vemuri, V. (1994). Adaptive User Interface for Intelligent Information Filtering. *Proceedings of the Third Golden West International Conference on Intelligent Systems*, pp 506-517.

Mock, Kenrick. (1993). A Genetic Classification System via Discrimination Tables. *Proceedings of the Third International Conference for Young Computer Scientists*.

Norcio, A. F. (1991). Adaptive Interfaces: Modeling Tasks and Users. *Proceedings of the 1991 IEEE International Conference on Systems, Man, and Cybernetics*, pp. 1099-1103.

Paice, C.D. (1989). Automatic Generation and Evaluation of Back-of Book Indexes. *Prospects for Intelligent Retrieval, Informatics 10*, Cambridge MA.

Peterson, J.L. (1982). *Webster's Seventh New Collegiate Dictionary: A Computer-Readable Format* (Technical Report TR-196). University of Texas at Austin, Department of Computer Science.

Quafafou, M. & Mock, K. (1995). Shadow: Adaptation of the Tutoring Interaction to the Changing Interests of the Student. *Seventh World Conference on Artificial Intelligence in Education.*

Quafafou, M. (1993). GAITS: Fuzzy Sets-Based Algorithms for Computing Strategies Using Genetic Algorithms. *Proceedings on Fuzzy Logic in Artificial Intelligence, FLAI* 1993, in The Lecture Notes in Artificial Intelligence, Springer Verlag.

Ram, A. (1992). Natural Language Understanding for Information-Filtering Systems. *Communications of the ACM*, **35** (12), pp. 80-81.

Resnick, P. et al. (1994). GroupLens: An Open Architecture for Collaborative Filtering of Netnews. *Internal Research Report, MIT Center for Coordination Science.*

Rich, E. (1979). User modeling via stereotypes. *Cognitive Science*, **3**, pp. 329-354.

Riecken, D. (1994). Intelligent Agents. *Communications of the ACM*, **37** (7), pp. 18-21.

Riesbeck, C. K. and Martin, C. E. (1986). Direct Memory Access Parsing. In. Kolodner, J. and Riesbeck, R. (Eds.), Experience, Memory, and Reasoning. Hillsdale, NJ: Lawrence Erlbaum.

Riesbeck, C. & Schank, R. (1989). Inside Case-based Reasoning. Hillsdale, NJ: Lawrence Erlbaum.

Riloff, E. & Lehnert, W. (1992). Classifying Texts Using Relevancy Signatures. *Proceedings of the Tenth National Conference on Artificial Intelligence*, Cambridge, MA: MIT Press, pp. 329-334.

Riloff, E. (1993). Using Cases to Represent Context for Text Classification. *Proceedings of the 1993 Spring Symposium on Case-Based Reasoning and Information Retrieval*, AAAI Press, pp. 90-97.

Rouse, W. B., Hammer, J.M., & Lewis, C. M. (1989). On capturing humans' skills and knowledge: Algorithmic approaches to model identification. *IEEE Transactions on Systems, Man, and Cybernetics*, **19**, 3, pp. 558-573.

Salton, G. The SMART Retrieval System: Experiments in automatic document processing. Englewood Cliffs, NJ: Prentice-Hall, 1971.

Salton, G. (1991). Developments in Automatic Text Retrieval. *Science*, 253, pp. 974-980.

Schank, R.C. & Abelson, R. (1977). Scripts, Plans, Goals, and Understanding. Hillsdale, NJ: Lawrence Erlbaum Associates, Publishers.

Schank, R.C. (1982). Dynamic memory. Cambridge, NY: Cambridge University Press.

Schank, R.C. (1992). Where's The AI? *AI Magazine*, **13**, pp. 38-49.

Scholtes, J.C. (1991). Unsupervised Context Learning in Natural Language Processing. *International Joint Conference on Neural Networks*, **1**, pp. 107-112.

Schwartz, M.F. (1992). A Comparison of Internet Resource Discovery Approaches. *Computing Systems*, **5** (4), pp. 461-493.

Shardanand, U. (1994). Social Information Filtering for Music Recommendation (Masters Thesis, Department of Computer Science and Engineering, Massachusetts Institute of Technology).

Sheth, B.D. (1994). A Learning Approach to Personalized Information Filtering (Masters Thesis, Department of Computer Science and Engineering, Massachusetts Institute of Technology).

Stadnyk, I. and Kass, R. (1992). Modeling Users' Interests In Information Filters. *Communications of the ACM*, **35** (12), pp. 49-50.

Stevens, C. (1992). Knowledge-Based Assistance for Accessing Large, Poorly Structured Information Spaces (Doctoral Dissertation, Department of Computer Science, University of Colorado).

Shick, A.G. Gordon, L.A., and Haka, S. (1990). Information Overload: A Temporal Approach. *Accounting Organizations and Society*, **15** (3), pp. 199-220.

Simon, H.A. (1981). The Sciences of the Artificial, Cambridge, MA:The MIT Press.

Thukral, V.K. (1983). Cognitive Strain as a Cause of Negative Bias, Ph.D. Dissertation Thesis. School of Business, University of Kansas, February 1983.

Voorhees, E. M. (1993). On Expanding Query Vectors with Lexically Related Words. *Proceedings of the Second Text REtrievel Conference (TREC-2)*, pp. 223-231.

Weiss, S.M. and Kulikowski, C.A. (1991). Computer Systems That Learn. San Mateo, CA: Morgan Kaufmann Publishers.

Yan, T.W. and Garcia-Molinia, H. (1994). Index structures for selective dissemination of information under the boolean model. *ACM Transactions on Database Systems*, **6**.

Yang, Y. (1994). An Example-Based Mapping Method for Text Categorization and Retrieval. *ACM Transactions on Information Systems*, **12** (3) pp. 252-277.